VŠB- Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra měřicí a řídicí techniky

Bakalářská práce

2010 **Otmar Masný**

VŠB- Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Katedra měřicí a řídicí techniky

Vestavěný řídicí systém pro aplikaci dynamického zobrazovače

Embedded system for dynamical monitoring

2010 **Otmar Masný**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra měřicí a řídicí techniky

Akademický rok 2008/2009

Zadání bakalářské práce

Student: **Otmar Masný**

Studijní program: B2645 Elektrotechnika, sdělovací a výpočetní technika

Studijní obor: 2612R041 Řídicí a informační systémy

Téma: Vestavěný řídicí systém pro aplikaci dynamického zobrazovače

Embedded system for dynamical monitoring

Zásady pro vypracování:

1. Rozbor problematiky digitálního zpracování signálu
2. Návrh systému pro dynamické zobrazování dat
3. Řešení systému dynamického zobrazení
4. Srovnání naměřených, analyzovaných výsledků s teoretickými předpoklady
5. Zhodnocení dosažených výsledků

Seznam doporučené odborné literatury:

1. Elektronika I, České Budějovice, 2003, ISBN 80-7232-171-4
2. Elektronika II, České Budějovice, 2003, ISBN 80-7232-212-5
3. Elektronika, Osadní 31, 170 00 Praha 7, 2002, ISBN 80-85970-42-2
4. Objektově orientované programování v C++, České Budějovice, 1995, ISBN 80-85828-20-0
5. Programujeme v jazyce C++, Praha, 1997, ISBN 80-85896-91-5
6. Microsoft Visual C# 2005 Krok za krokem, Brno, 2006, ISBN 80-251-1156-3
7. Elektronika- obvody, součástky, děje, Praha, 1998, ISBN 80-86056-25-2
8. Příručka pro elektrotechnika, Praha, 2002, ISBN 80-86706-00-1
9. Přehled diskrténích polovodičových součástek TESLA, Praha, březen 2002, ISBN 80-7300-061-X
10. V. KADLEC. Učíme se programovat v jazyce C. Computer press, Praha, 2002,ISBN 80-7226-715-9

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Zdeněk Macháček, Ph.D.**

Datum zadání: 30.11.2008

Datum odevzdání: 07.05.2010

doc. Ing. Jiří Koziorek, Ph.D.
vedoucí katedry

prof. Ing. Ivo Vondrák, CSc.
děkan fakulty

Prohlášení:

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

…….
Otmar Masný

Datum odevzdání bakalářské práce: 20. srpna 2010

Poděkování:

Tímto bych chtěl poděkovat panu ing. Zdeňkovi Macháčkovi Ph.D. za jeho cenné odborné rady a konzultace.

Dále pak děkuji Richardovi Adamovskému za spolupráci na uživatelském editačním softwaru a za další pomoc a rady při programování.

Můj dík patří také všem ostatním, kteří mně jakýmkoliv způsobem inspirovali, pomáhali a podporovali.

Abstrakt

Tato bakalářská práce se zabývá popisem návrhu a samotnou realizací řídicího systému dynamického zobrazovače. Dynamický zobrazovač využívá ke své funkci především nedokonalých vlastností lidského zraku. Dále pak pracuje s číslicovými signály. První část bakalářské práce tedy obsahuje teoretické poznatky, které jsou nezbytné pro správnou funkci zobrazovače. Celý systém dynamického zobrazovače se skládá z několika podsystémů, které jsou popsány v části následující jak z teoretického hlediska, tak z hlediska praktického pro celý řídicí systém. V poslední části této práce je praktický popis samotné realizace celého dynamického zobrazovače. Možné využití se nabízí v komerční elektrotechnice.

Klíčová slova

ATmega16, Atmel, AVR Studio, Dynamický zobrazovač, Eagle, EEPROM paměť, FT232RL konvertor, Hallův snímač, MCU, Microsoft Visual Studio, mikroprocesor, obrazec, PWM modulace, RS232, Shanonův teorém, USB sběrnice, vzorky

Abstract

This bachelor thesis describes the design and actual implementatiton of control system for dynamical monitoring. Dynamical monitoring used to function mainly imperfect properties of human vision. Then work with digital signals. The first part of this bachelor thesis contains the theoretical knowledge that are necessary for proper function of the dynamical monitoring. The whole system is a dynamical monitoring is composed of several subsystems, which are described in the following part both in theory and in term of the whole control system. In the last part of this work is the description of the practical realization of the dynamical monitoring. Possible uses are offered in commercial electrical engineering.

Key words

ATmega16, Atmel, AVR Studio, Dynamical Monitoring, Eagle, EEPROM memory, Figure, FT232RL converter, Hall sensor, MCU, microprocessor, PWM modulation, Shannon theorem, USB, RS232, Samples

Seznam použitých zkratek

ALU	Arithmetic logic unit
AVR	Označení pro mikroprocesory Atmel
CISC	Complex Instruction Set Computer
CMOS	Complementary Metal–Oxide–Semiconductor
CTRL	Control
CTS	Clear to Send
DCD	Data Carrier Detect
DIS	Discharge
DPLL	Davis Putnam Logemann Loveland
DPS	Deska plošného spoje
DSR	Data Set Ready
DTR	Data Terminal Ready
EAGLE	Easily Applicable Graphical Layout Editor
EEPROM	Electrically Erasable Programmable Read- Only Memory
FIFO	First In First Out
FTDI	Future Technology Devices International
GND	Ground
ICP	Input Capture pin
INT0	Interrupt 0
LDO	Low Drop- Out
LED	Light Emitting Diode
MCU	Micro Controller Unit
NRZI	Non Return To Zero Invert
PC	Personal Computer
PWM	Pulse width Modulation
RFID	Radio Frequency Identification
RI	Ring Indicator
RISC	Reduced Instruction Set Computer
RS232	Recomended Standard 232
RTS	Request to Send
RxD	Receive Data
SGND	Signal Ground
SIE	Serial Interface Engine
TRH	Threshold
TRIG	Trigger
TxD	Transmit Data
UART	Universal Asynchronous Reciever and Transmitter
UPE	USB Protocol Engine
USART	Universal Synchronous and Asynchronous Reciever and Transmitter
USB	Universal Serial Bus
WiMAX	Worldwide Interoperability for Microwave Access

Seznam použitých symbolů

B	Magnetická indukce {T}
B_{OP}	Mag. indukce pro sepnutí {T}
B_{RP}	Mag. indukce pro vypnutí {T}
C	Konstanta
E	Energie signálu
F	Síla působící na vodič {N}
I	Elektrický proud {A}
I_p	Elektrický proud polovodičem {A}
I_{LED}	El. proud protékající LED {A}
P	Střední výkon signálu
$P_{A,B}$	Vzájemný výkon dvou signálů
R_H	Hallova konstanta
R_v	Odpor vinutí kotvy {Ω}
R_{LED}	Odpor rezistoru u LED {Ω}
T	Perioda {s}
T_S	Vzorkovací perioda {s}
U_{CC}	Napájecí napětí {V}
U_i	Indukované napětí kotvy {V}
U_H	Hallovo napětí {V}
U_{LED}	Elektrické napětí na LED {V}
U_{RLED}	El. napětí na rezistoru u LED {V}
W_{MAX}	Amplituda signálu
d	Tloušťka pol. materiálu {m}
f	Frekvence {Hz}
f_d	Frekvence děličky rezonátoru {Hz}
f_r	Frekvence rezonátoru {Hz}
f_S	Vzorkovací frekvence {Hz}
f_{MAX}	Max. vzorkovací frekvence {Hz}
k	Počet vzorků
l	Délka vodiče {m}
n	Počet otáček motoru
n_i	Počet impulsů na jednu otáčku
q	Elementární náboj
t	Čas {s}
t_i	Doba jednoho impulsu {s}
t_n	Doba jedné otáčky {s}
t_s	Doba vykreslení sloupce {s}
$w(t)$	Signál se spojitým časem
$w[k]$	Signál s diskrétním časem
$w_I(t)$	Impulsní signál
$v_I(t)$	Vytvořovací signál
w_{STR}	Střední hodnota signálu
w_{EF}	Efektivní hodnota signálu
Φ	Magnetický indukční tok {Wb}
δ	Diracův impuls
η_q	Hustota nosičů náboje
φ	Fázový posun {rad}
ω_s	Vzorkovací úhlová rychlost {rad/s}

Obsah

1 Úvod

Cílem této bakalářské práce je navrhnout a realizovat vestavěný řídicí systém pro aplikaci dynamického zobrazovače. Dynamický zobrazovač má za úkol vykreslovat texty a obrazce. Celý systém se skládá z několika částí, které byly navrženy tak, aby celá aplikace splňovala dané požadavky.

Nejdůležitější částí systému je část zobrazovací, která je tvořena sloupcem 16- ti LED, z nichž každá tvoří jeden zobrazovací bod. Tyto zobrazovací body využívají setrvačnost lidského zraku podobně jako televizní technika nebo kinematografie. Jednotlivé obrazce nebo texty jsou tedy postupně vykreslovány pomocí tohoto jediného sloupce LED tak, že příslušné zobrazovací body ve sloupci svítí určitý čas potřebný k vykreslení jednoho úseku textu nebo obrazce. Po vykreslení jednoho úseku se začne vykreslovat úsek další až je postupně vykreslen celý obrazec. Ovšem k tomu je zapotřebí aby vykreslovací sloupec s LED rotoval, protože by se jinak jednotlivé zobrazené úseky obrázku překrývaly a nebylo by dosaženo požadovaného efektu. Proto je celá řídicí i zobrazovací elektronika roztáčena stejnosměrným motorem, který v podstatě vytváří obrazovku. Protože celá elektronika musí rotovat, bylo nutné vytvořit systém napájení, který by byl schopen aplikaci napájet. Otáčky motoru jsou regulovány pomocí jednoduché PWM.

Vykreslovací data jsou digitální a jsou vytvořena pomocí editačního softwaru. Tento software vytváří uživatelské rozhraní pro PC, ve kterém lze přímo definovat texty a obrázky, které se budou vykreslovat. V podstatě převádí napsaný text nebo nakreslený obrázek do digitálních hodnot, které jsou pak z PC pomocí tohoto softwaru odesílány do dynamického zobrazovače.

Pro odesílání již editovaných dat z PC byla zvolena sběrnice USB, a to především kvůli své velké univerzálnosti a rozšířenosti. V samotné aplikaci dynamického zobrazovače jsou ale data přijatá ze sběrnice USB převedena na RS232 pomocí dnes velmi populárního převodníku FT232RL, který využívá virtuální sériový port. Tato komunikace je pouze jednosměrná.

Takto odeslaná a převedená data jsou následně uložena v EEPROM mikroprocesoru Atmel ATmega16. Mikroprocesor tyto přijatá dat následně zpracovává a posílá na zobrazovací část tvořenou LED. Mikroprocesor také aktualizuje čas potřebný k rozsvícení příslušných zobrazovacích bodů ve sloupci LED.

Tento čas je odvozen z otáček stejnosměrného motoru pro zřetelný a synchronizovaný obraz. Otáčky motoru jsou snímány pomocí Hallovy sondy.

2 Vlastnosti lidského zraku

Tato bakalářská práce se zabývá zobrazováním dat podobně jako filmová a televizní technika, která využívá jistých nedokonalostí lidského zraku ke své funkci, stejně jako tato bakalářská práce.

V této kapitole jsou tedy popsány základní vlastnosti lidského zraku.

2.1 Lidský zrak

Je to složitý systém propojení mezi nervovými buňkami zrakového systému na různých úrovních zpětných vazeb.

Lidské oko je tvořeno dvěma základními systémy:

- Optický systém- slouží k vytváření převráceného, zmenšeného a neskutečného (virtuálního) vnějšího světla na sítnici oka

  Optický systém je složen z:
  - rohovky- přední části oka (průhledná vnější vrstva)
  - duhovky- barevná část oka
  - zornice- otvor, jež se podle potřeby rozšiřuje nebo zužuje od 2 mm do 8mm (černý střed duhovky)

- Nervový systém- slouží k příjmu a zachovávání zrakové informace

  Nervový systém je složen z:
  - sítnice- 11 vrstev buněk
  - tyčinek- snímače pro noční, neboli černobílé- skotopické vidění (v jednom oku jich je obsaženo asi 128 miliónů)
  - čípků- snímače pro denní, neboli fotopické vidění (průměrný člověk jich má kolem 6,5 miliónů)

2.2 Fyziologické vlastnosti lidského oka

Rychlost vnímání (setrvačnost)

Při dopadu světla na sítnici oka dojde ke dráždění foto-citlivých receptorů, což je ve zrakovém centru mozku vyhodnoceno jako zrakový vjem. Po zániku fyzikálního popudu tento zrakový vjem doznívá tak dlouho, dokud nedojde k regeneraci zrakového purpuru, což trvá obvykle 1/7 až 1/3 sekundy. Při rychlých periodických změnách jasu nedojde k zániku předchozího vjemu, a proto jednotlivé zrakové vjemy splynou.

Potřebná frekvence změn ke splynutí vjemů závisí na jasu obrazu a podrážděné části sítnice. Zvětšením kmitočtu impulsů nad 13 za vteřinu nebo zkrácením přestávek mezi nimi je dosaženo stálé intenzity zrakových vjemů a oko vnímá střední hodnotu jasu.

Díky této nedokonalosti lidského oka vznikla kinematografie a televizní technika, neboť u obou je využito promítání dílčích obrazových snímků v rychlém sledu za sebou. Na stejném principu je založena i tato bakalářská práce, kde každá otáčka řady LED znamená promítnutí obrázku. Takže při vyšší hodnotě otáček je tento obrázek promítán v rychlém sledu za sebou, čímž je dosaženo jeho celkového splynutí a pozorovatel vidí celý obrázek jasně.

Rozlišovací schopnost

Rozlišovací schopnost oka je schopnost oka rozeznat na určitém pozadí dva detaily. Běžná rozlišovací schopnost lidského oka je 1´, což je dalším faktorem přispívajícím k tvorbě filmového obrazu.

Filmový či televizní obraz je tvořen souborem bodů a řádků. Absolutně kompaktní tedy bude pouze v případě, že jejich body jsou na obrazovce rozmístěny pod úhlem menším než je 1´.

Tento faktor má vliv i na konstrukci této bakalářské práce, neboť čím blíže jsou zobrazovací LED u sebe, tím kompaktnější je celkový obraz.

Další vlastnosti lidského oka:

- akomodace- schopnost oka zaostřit na sítnici vzdálenosti, které jsou kratší než 6 m (nejbližší bod, které je oko ještě schopno zaostřit)
- adaptace- schopnost oka přizpůsobit se různým intenzitám osvětlení (rozmezí při barevném vidění je od 0,25 lx až 10^5 lx a při černobílém vidění až do 10^{-9} lx)
- zorné pole- část prostoru ohraničená horizontálně, kterou může pozorovatel postřehnout upřeným pohledem bez zapojení očních svalů a svalů hlavy (rozpětí vertikálně je 6° a horizontálně je 8°)
- obhledové pole- část prostoru, kterou může pozorovatel sledovat zapojením očních svalů a svalů hlavy
- zraková ostrost- schopnost oka zhodnotit jasnost odlišovaných detailů
- prostorové vidění- stereoskopický mechanismus založený na základě individuálních zkušeností (princip superpozice překrývání)

[1]

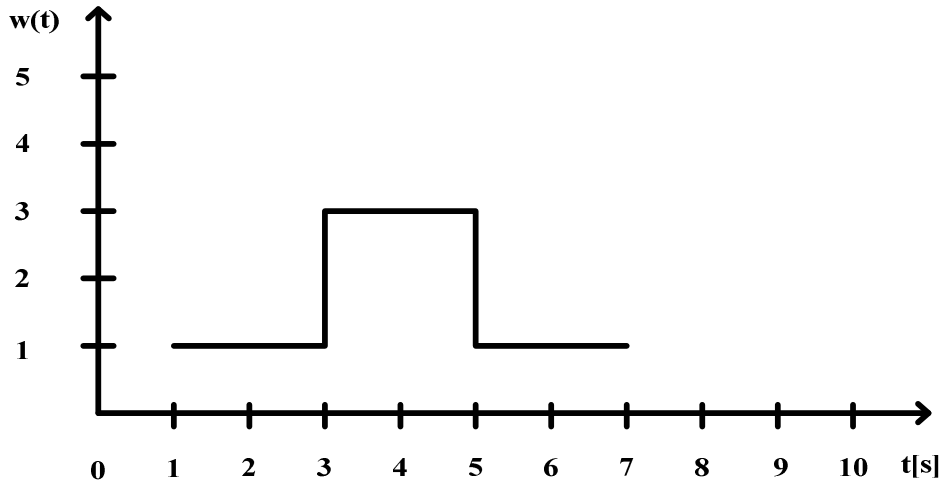
3 Základní popis a rozdělení signálů

Tato kapitola obsahuje základní přehled a popis signálů, se kterými se běžně setkáváme, zvláště pak signálů digitálních, resp. signálů s diskrétním časem, protože je dynamický zobrazovač využívá ke své činnosti.

3.1 Rozdělení signálů podle charakteru

3.1.1 Signály se spojitým časem

Signály se spojitým časem jsou definovány v každém bodě časového průběhu $t \in \Re$, nabývající reálných hodnot $w(t) \in \Re$ jak je vidět na Obr. 1 .



Obr. 1 Příklad signálu se spojitým časem

[2]

3.1.2 Signály s diskrétním časem

Důležitou operací v řadě aplikačních oblastí jako jsou komunikace a samočinné řízení je vzorkování signálu se spojitým časem $w(t)$. Při standardním vzorkování jsou vzorky hodnot signálu $w(t)$ rovny $w(k \cdot T_S)$, kde k může nabývat hodnot $k = 0, \pm 1, \pm 2, ...$ Pro zjednodušení matematické analýzy procesu vzorkování analogového signálu $w(t)$ se obvykle předpokládá, že lze vzorkovaný signál vyjádřit ve tvaru součinu signálu se spojitým časem a signálu tvořeného časovou posloupností Diracových impulsů daných sumou(tzv. vytvořovací signál).

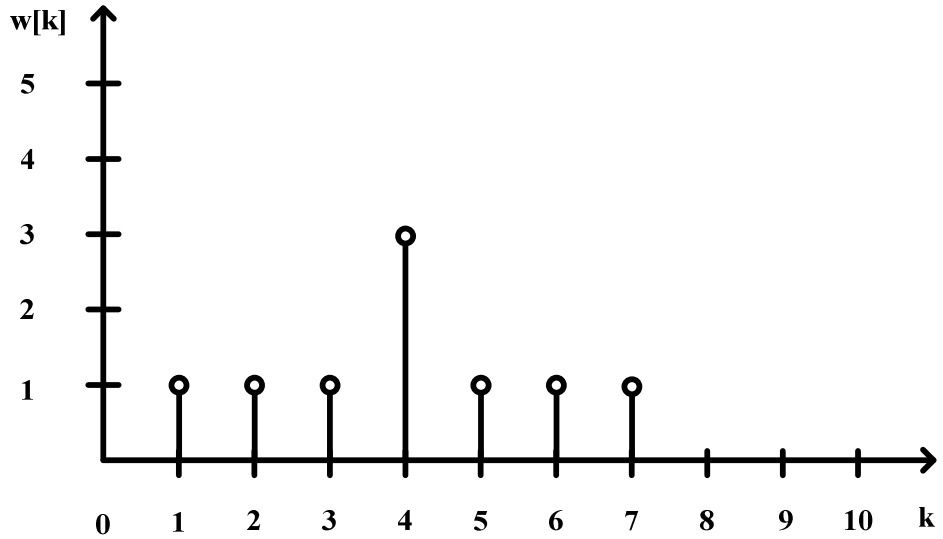
$$w_I(t) = w(t) \cdot v_I(t) = \sum_{k=-\infty}^{k=+\infty} w(t)\delta(t - kT_S)$$

Takto navzorkovaný signál $w_I(t)$ tedy nazýváme signálem impulsním, který má tvar časové posloupnosti Diracových impulsů, jejichž mohutnost je dána hodnotami $w(k \cdot T_S)$ signálu $w(t)$. Mimo tyto body má impulsní signál nulovou hodnotu.

Ovšem signály s diskrétním časem jsou definované pouze v určitých diskrétních okamžicích časového průběhu. Mimo tyto diskrétní časové okamžiky nejsou definovány. Abychom obdrželi signál s diskrétním časem $w[k]$, jednoduše v impulsním signálu se spojitým časem $w_I(t)$ zvolíme $t = k \cdot T_S$ pak

$$w[k] = w_I(k \cdot T_S)$$

Příklad signálu s diskrétním časem je na Obr. 2.



Obr. 2 Příklad signálu s diskrétním časem

Aby byly výsledky analýzu signálu s diskrétním časem interpretovatelné, musí vzorkování spojitého signálu $w(t)$ splňovat podmínky tzv. Shanonova a Kotělníkova teorému, který zní: Přesná rekonstrukce spojitého, frekvenčně omezeného signálu z jeho vzorků je možná tehdy, pokud byl vzorkován frekvencí alespoň dvakrát vyšší, než je maximální frekvence rekonstruovaného signálu. Matematický zápis Shanonova teorému je tedy:

$$f_s \succ 2 \cdot f_{\max}$$

$$\omega_s \succ 2 \cdot \omega_{\max}$$

Při nedodržení Shanonova a Kotělníkova teorému dochází k jevu zvaném Aliasing. Prakticky dochází ke překrytí frekvenčních spekter vzorkovaného signálu a tedy ke ztrátě informace.Typickým příkladem je snímání rychle rotujících předmětů (vrtule letadla, kolo rychle jedoucího automobilu, apod.) kamerou. Projevuje se tím, že divák vidí kolo nebo vrtuli

otáčet se opačným směrem, než se má ve skutečnosti otáčet nebo se pohybují pomalu. K tomu, aby se vrtule otáčela správně by musela kamera zachytit alespoň dva snímky za jednu otáčku.

Aliasing lze zpozorovat také u dynamického zobrazovače při nízkých otáčkách motoru, protože se nebude obrazec promítat dostatečně rychle za sebou, aby z nich pak vyplynul celý. Důsledkem toho pak vykreslovaný obrazec může blikat, protože je jeho promítání pomalé.

Charakteristické veličiny signálů s diskrétním časem

- Střední hodnota signálu $w[k]$: $w_{STR} = \lim_{K\to\infty} \frac{1}{K} \sum_{k=-K/2}^{k=+K/2} w[k]$
- Energie signálu $w[k]$: $E = \sum_{k=-\infty}^{k=+\infty} w^2[k]$
- Střední výkon signálu $w[k]$: $P = \lim_{K\to\infty} \frac{1}{K} \sum_{k=-K/2}^{k=+K/2} w^2[k]$
- Efektivní hodnota signálu $w[k]$: $w_{EF} = \sqrt{P}$
- Vzájemný výkon signálů $w_A[k]$ a $w_B[k]$: $P_{AB} = \lim_{K\to\infty} \frac{1}{K} \sum_{k=-K/2}^{k=+K/2} w_A[k] \cdot w_B[k]$

[3], [4]

3.1.3 Diskrétní (číslicový) signál

Číslicové signály jsou definované pouze v určitých bodech časového průběhu, které jsou řazeny v posloupnosti celočíselných násobků s danou periodou $k \cdot T_s$, nabývající pouze omezený konečný počet hodnot $w[k]$.

[2]

3.2 Rozdělení signálů podle jejich průběhu

- výkonové signály- signály s nenulovým, konečným středním výkonem $P \neq 0, P \prec \infty$
- energetické signály- signály s nenulovou konečnou energií, $E \neq 0, E \prec \infty$
- harmonické signály- periodicky se opakující signály harmonického průběhu (patří mezi výkonové)
- periodické signály- periodicky se opakující signály s periodou T (patří mezi výkonové)
- náhodné signály- signály s náhodným průběhem (šumy)
- konstantní signály- signály s konstantním průběhem

[2]

4 Návrh řídicího systému

Na Obr. 3 je znázorněno blokové schéma návrhu řídicího systému Dynamického zobrazovače, které demonstruje celou funkčnost řídicího systému. K tomu, aby mohl Dynamický zobrazovač pracovat musí nejprve obdržet data, která se mají vykreslit. Tyto data jsou vytvořena mimo celý model v PC pomocí uživatelského programu pro editování. Tento software umožňuje uživateli vytvořit text nebo vložit obrázek, který chce následně vykreslit. Takto vytvořený text nebo obrázek se následně „navzorkuje“ na digitální hodnoty, které jsou uloženy do souboru. Takto vytvořený soubor se pak pomocí tohoto programu již může odeslat do zařízení.

Každý modernější PC obsahuje již v dnešní době USB konektory pro přijímání a odesílání dat. Proto i řídicí systém byl navrhnut tak, aby bylo možné odesílat softwarově editovaná data z PC do aplikace pomocí sběrnice USB. Nejjednodušším způsobem jak toho dosáhnout je použití USB převodníku na RS232, který využívá virtuální COM port, čímž prakticky vznikne komunikační rozhraní USB/ RS232. Pomocí toho rozhraní a editačního softwaru jsou tedy vzorky dat uložené do souboru odeslány do EEPROM, která je součástí MCU. Toto komunikační rozhraní je pouze jednosměrné, protože ke správné funkci je zapotřebí pouze přijímání dat z PC.

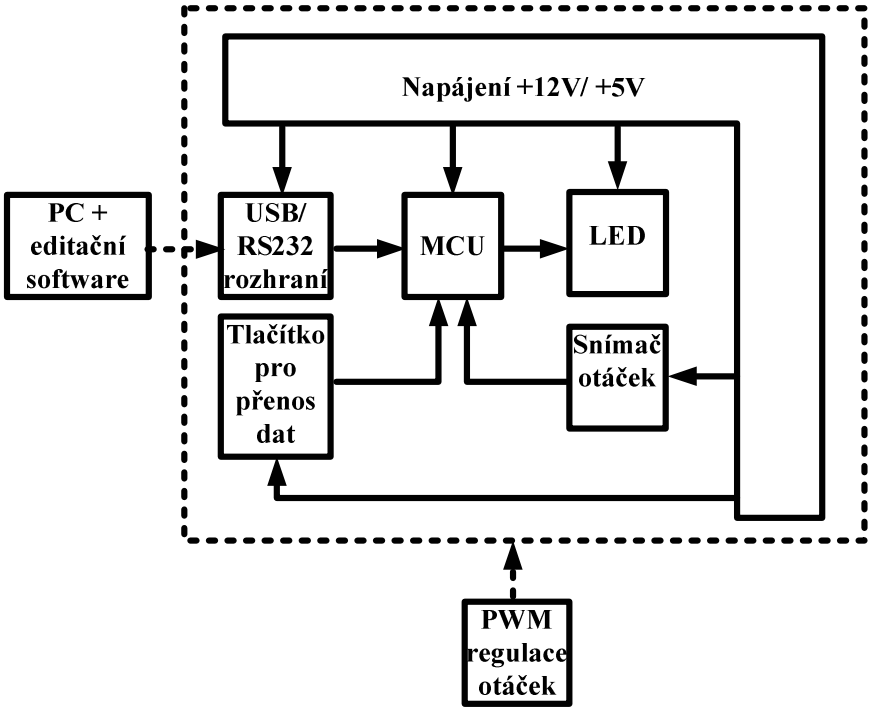
Aby nedocházelo k chybám, byla aplikace doplněna o tlačítko, kterým se zahajuje celý proces odesílání dat. Tedy po stisknutí tohoto tlačítka bude MCU teprve přijímat data, ne jinak.

Po obdržení editovaných dat již pak MCU čeká na to, až uživatel uvede do chodu stejnosměrný motor, který otáčí celou elektronikou. Až se tak stane, tak na základě údaje o velikosti otáček motoru, které zprostředkovává Hallova sonda MCU vysílá na své výstupy data, která byla před tím uložena do EEPROM paměti.

Hallova sonda zde plní důležitou funkci, neboť pouze díky aktualizovaným informacím o velikosti otáček stejnosměrného motoru je mikroprocesor schopen vypočítat časovou konstantu pro zobrazení jednoho vzorku vykreslovaného obrázku a také stabilizovat celkový obraz tak, aby nedocházelo k jeho deformaci.

Výstupní data z MCU jsou zobrazována již pomocí LED diod, které jsou těmito výstupními daty spínány. Pokud se tedy motor točí dostatečně rychle, může uživatel sledovat text nebo obrázek, který zadal prostřednictvím programu pro uživatelské rozhraní.

Celý systém je napájen stejnosměrným napětím +12V, které je následně stabilizátorem sníženo na +5V. Napětí +5V pak napájí celou aplikaci. To ovšem neplatí o PC a PWM regulaci otáček, které jsou napájeny z jiných zdrojů elektrické energie.



Obr. 3 Blokové schéma elektroniky řídicího systému

5 Využití snímače otáček v Dynamickém zobrazovači

Pro správné zobrazování dat v Dynamickém zobrazovači je nutné, aby byl výsledný obraz stabilní a nedocházelo během zobrazování k jeho deformaci. Zároveň je nutné určit meze, ve kterých bude výsledný obrazec vykreslen.

K tomu byl použit snímač otáček. Jak již bylo uvedeno, celá řídicí elektronika i se zobrazovací částí rotuje díky stejnosměrnému motoru. Motor tudíž vytváří pomyslnou obrazovku a také díky snímači jeho otáček je řídicí systém schopen vypočítat časovou konstantu potřebnou pro vykreslení jednoho úseku vykreslovaného obrazce. Vypočtená časová konstanta se tedy odvozuje od otáček stejnosměrného motoru, kde doba jedné otáčky odpovídá době potřebné pro vykreslení celého obrazce. Pokud se tedy bude motor točit dostatečně rychle, nebude lidské oko schopné jednotlivé obrazce jdoucí za sebou rozeznat a výsledný obraz bude věrohodný.

Hlavním důvodem použití snímače otáček stejnosměrného motoru je tedy synchronizace obrazu a správný výpočet časové konstanty pro vykreslení jednoho sloupce.

5.1 Úloha a postavení snímače v měřicích a řídicích systémech

Informace jsou mimo jiné jednou z největších potřeb dnešní lidské společnosti a na základě této potřeby vznikly Informační systémy, jejichž smyslem je identifikace informace, její následné zpracování a využití.

Podskupinu obecných informačních systémů- systému pro zpracování informací představují Elektronické měřicí a řídicí systémy (Měřicí a řídicí systémy).

Každý systém pro zpracování informací (a tedy i měřicí a řídicí systém) se skládá ze tří částí: vstupního převodníku, bloku pro vlastní zpracování informací a výstupního převodníku.

Ve vstupním převodníku dochází k identifikaci informace o stavu měřené resp. regulované soustavy a jejímu převodu na formu, která je vhodná pro zpracování informace v následujícím bloku.

V současné době je převažující formou pro účely zpracování informace elektrický signál. Vstupní převodník (senzor, snímač) představuje primární zdroj informace o měřené nebo řízené soustavě a určuje vlastnosti celého měřicího popř. řídicího systému. Ve většině případů i dnes tvoří nejnákladnější část celého řetězce.

[5]

5.2 Snímání otáček Hallovým senzorem

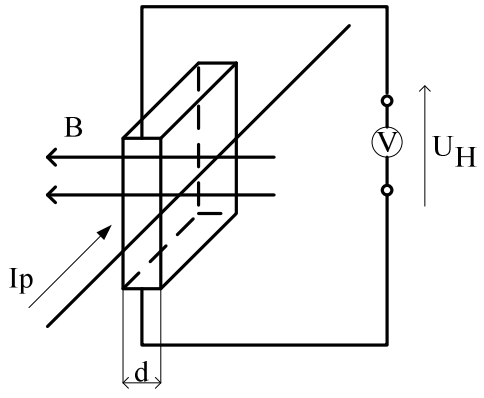
Jako senzor pro snímání otáček stejnosměrného motoru Dynamického zobrazovače byl použit Hallův senzor především díky jednoduchosti jeho použití a dobrých vlastností.

5.2.1 Hallův jev

Hallův jev spočívá v tom, že příčné magnetické pole působí na polovodičový materiál, jímž prochází el. proud tak, že na protilehlých stranách ve směru kolmém na procházející proud se indukuje Hallovo napětí (viz také

Obr. 4):

$$U_H = R_H \cdot \frac{I_p}{d} \cdot B \text{, kde } R_H = \frac{3}{8} \cdot \frac{1}{\eta_q \cdot q}$$



Obr. 4 Princip Hallova jevu

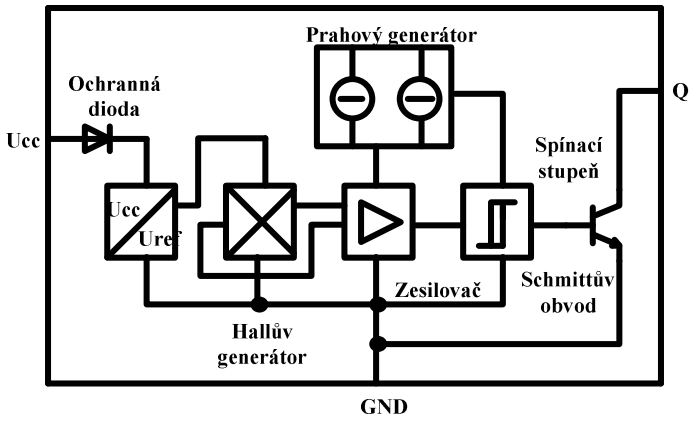
[5]

5.2.2 Hallův senzor TLE4905L

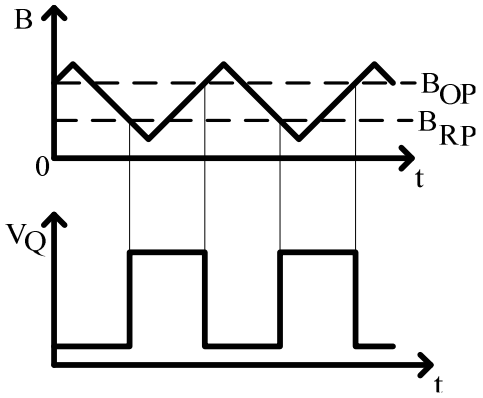
Jedná se o unipolární Hallův senzor, kde je na jednom čipu integrován referenční zdroj napájení, Hallův generátor, zesilovač a Schmittův klopný obvod, což je velmi výhodné, protože výstupní signál z tohoto senzoru má dostatečnou kvalitu a díky tomu nedochází ke zkreslení informace.

Magnetické pole působící kolmo k povrchu čipu vyvolává Hallovo napětí, které je následně zesíleno a přivedeno na kolektorový výstup, který je chráněn proti rušení. Součástí senzoru je i ochranná dioda proti přepólování napájecího napětí. Blokové schéma čipu TLE4905L je na Obr. 5.

Tento typ senzoru je unipolární, což znamená, že pokud je magnetická indukce pozitivní (reaguje pouze na jeden pól přiloženého magnetu) a je překročena její zapínací hodnota, tak se na výstupu objeví log. 0. Pokud ovšem hodnota magnetické indukce klesne pod minimální mez, tak bude na výstupu senzoru neustále log. 1, dokud nedojde k opětovnému překročení zapínací hodnoty magnetické indukce. Při návrhu řídicího systému je tedy směrodatná sestupná hrana výstupního signálu z Hallova snímače. Na Obr. 6 je zobrazen teoretický výstupní signál.



Obr. 5 Blokové schéma zapojení Hallova senzoru TLE4905L



Obr. 6 Výstupní signál Hallova senzoru TLE4905L

[6]

6 Komunikační rozhraní USB a RS232

Pro nejvhodnější způsob komunikace mezi editačním softwarem v PC a samotnou aplikací byl vybrán konektor USB pro svou velkou rozšířenost, univerzálnost a spolehlivost. Ovšem data odeslaná z PC do zařízení přes USB sběrnici jsou následovně převedena na RS232 obvodem FT232RL a pak dále odesílána do MCU pomocí virtuálního COM portu, který je nainstalovaný v konvertoru.

6.1 Rozhraní USB

- sériové rozhraní
- možnost připojování zařízení na relativně velkou vzdálenost (až 5 m)
- možnost napájet zařízení přímo z konektoru (možný odběr 100 až 500 mA)
- velký počet připojitelných zařízení (při použití HUBu až 127)
- podpora Plug&Play (připojování a odpojování zařízení za provozu)

[7]

6.1.1 Konektory a kabely USB

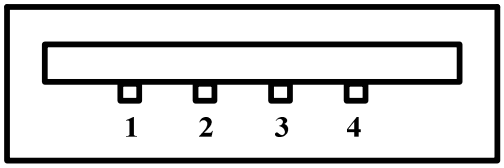
USB je sériová sběrnice, na které jsou data přenášena jednotlivě bit po bitu diferenčně (kvůli snížení rušení). Datové vodiče přenášejí vzájemně negované informace o napěťových úrovních 0 až 3,3V. Význam jednotlivých vodičů jsou v Tab. 1.

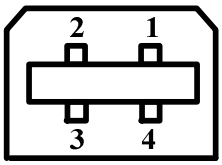
USB používá dva typy konektorů se čtyřmi vodiči. Jedná se o konektor typu A a o konektor typu B. Tyto konektory jsou vyráběny jak pro propojovací kabely, tak pro umístění na DPS. Pro oba druhy USB konektorů existují i zmenšené verze USB mini a USB mikro. Konektory typu A i B jsou vyobrazeny na Obr. 7 a Obr. 8.

Osobní počítače mají zabudovány USB konektory typu A. Menší zařízení (zejména počítačové myši) používají pevně připojený kabel s konektorem typu A. Rychlé zařízení s odnímatelným kabelem (tiskárny) používají konektor typu B, a proto se k počítači připojují pomocí kabelu, ve kterém jsou obsaženy oba typy USB konektorů, tedy A-B. Tento typ kabelu je také nejpoužívanějším. Existuje také kabel typu A-A, ovšem ten bývá používán je málokdy (např. spojení dvou počítačů apod.). USB propojovací kabely jsou vyráběny i pro verze USB mini a USB mikro.

Číslo vývodu	Význam vývodu
1	V_{CC} +5V (Napájecí napětí)
2	Data+ (Přímá data)
3	Data- (Negovaná data)
4	GND (zem)

Tab. 1 Čísla vývodů a jejich význam v USB konektoru





Obr. 7 USB konektor typu A Obr. 8 USB konektor typu B

[7]

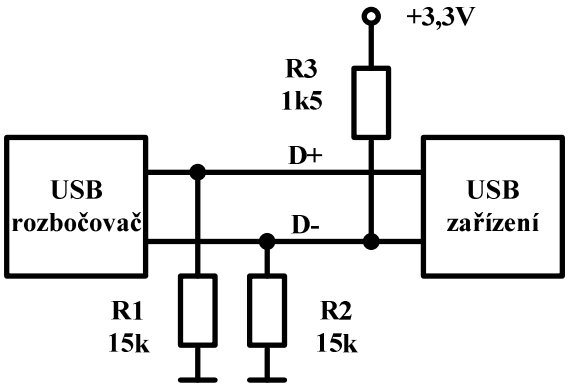
6.1.2 Verze USB a jejich přenosové rychlosti

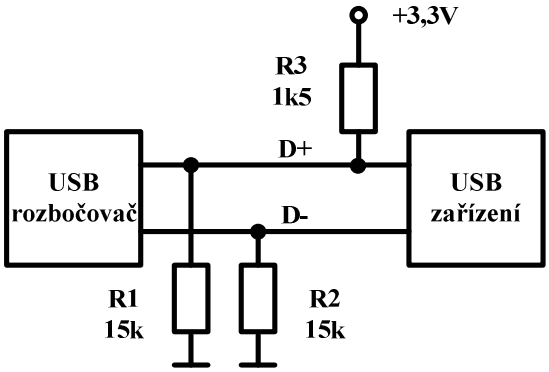
Zařízení USB pracují buďto ve starší verzi 1.1 nebo v novější verzi 2.0. Obě se od sebe odlišují především přenosovými rychlostmi, které jsou zaznamenány v Tab. 2.

Veškeré uvedené přenosové výkony verzí USB platí ovšem vždy pro připojení jednoho zařízení. Pokud je k počítači připojeno více zařízení, tak je celková šíře pásma přenosového výkonu rozdělena mezi jednotlivá zařízení. Přenosová rychlost je stanovena samotnými zařízeními pomocí pull- up rezistoru jak ukazují Obr. 9 a Obr. 10 Použitím pull- up rezistoru je zároveň podávána informace, že je připojeno zařízení.

Rychlost	Přenos	USB standard
Low Speed	1,5 Mb/s	USB 1.1/2.0
Full Speed	12 Mb/s	USB 1.1/2.0
High Speed	480 Mb/s	USB 2.0

Tab. 2 Verze USB a jejich přenosové rychlosti





Obr. 9 Zařízení Low speed Obr. 10 Zařízení Full speed nebo High speed

[7]

6.1.3 Přenos dat v USB

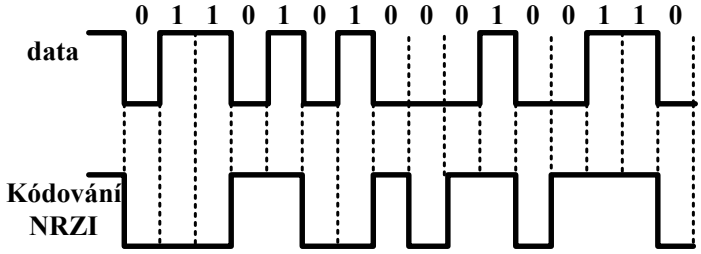
Přenos dat v USB je realizován pomocí přesně definovaných rámců délky přesně 1 ms, uvnitř kterých jsou postupně zapracovávány pakety o délce 8 až 64 bajtů pro několik zařízení.

Protože USB je jednomasterová sběrnice, tak jsou počítačem řízeny veškeré aktivity, tudíž žádné zařízení nemůže vysílat data samo od sebe. Je- li připojeno více zařízení, zajišťuje rozdělování paketů rozbočovač (HUB).

Podřízená zařízení Slave jsou synchronizována na tok dat. Hodiny přenosu jsou získávány přímo z datového signálu (hodinový signál není přenášen po speciální lince) a to metodou NRZI, kde nulové úrovně v datech jsou vedeny ke změně a úrovně jedniček jsou ponechány beze změn jak zobrazuje Obr. 11.

Samotný tok dat se děje mezi vysílačem a přijímačem. Přijímač musí být schopen získávat hodinový signál, přijímat a detekovat data. Pokud je v datovém toku zahrnuto šest po sobě jdoucích jedniček, tak je vysílačem přidána jedna nula navíc k vynucení změny úrovně pro obnovení hodinového kmitočtu (bit- stuffing). Přijímačem je pak tato navíc přidaná nula odstraněna (bit- unstuffing).

V zařízení je také obsažena jednotka SIE pro konverzi USB dat. Pro výměnu dat mezi jednotkou SIE a zbytkem zařízení jsou použity vyrovnávací paměti FIFO, které pracují jako posuvné registry. Paměti FIFO umožňují sladit rychlosti USB sběrnice a USB zařízení.



Obr. 11 Princip NRZI kódování

[7]

6.1.4 Druhy přenosů dat USB

- Řídicí přenos (Control Transfer)- řízení Hardware- vysoká priorita, automatické zabezpečení chyb
- Přenos pomocí přerušení (Interrupt Transfer)- periodické dotazování počítače na další data (myš, klávesnice)
- Hromadný přenos (Bulk Transfer)- přenos velkých množství dat se zabezpečením, nízká priorita přenosu (tiskárny, skenery)
- Isochronní přenos (Isochronu Transfer)- přenos velkých množství dat bez zabezpečení (zvuková karta)

[7]

6.1.5 Rozpoznávání USB zařízení

USB rozhraní podporuje tzv. Plug&Play, takže každé zařízení které bylo připojeno k počítači je automaticky rozpoznáno operačním systémem.

Samotné rozpoznání zařízení spočívá v tom, že se operační systém dotáže zařízení na určité parametry. Po připojení zařízení je pomocí HUBu rozpoznáno nové zařízení aktivací linky D+ nebo D-, při čemž jsou provedeny následující kroky:

- HUB informuje počítač o tom, že je připojeno zařízení
- počítač se dotáže HUBu, na který port je zařízení připojeno
- následně je tento port aktivován a je proveden reset USB sběrnice
- po nulovacím signálu o délce 10 ms je uvolněn proud 100 mA pro zařízení a jednotkou SIE je resetován mikrokontrolér a tak je zařízení připraveno
- počítačem jsou čteny první bajty deskriptoru zařízení ke stanovení délky datových paketů
- přiřazení sběrnicové adresy
- načtení informací z deskriptoru zařízení
- přiřazení jedné z možných konfigurací

[7]

6.1.6 HUB- USB rozbočovač

HUBy jsou používány pro připojení více zařízení k jednomu portu počítače. Maximálně lze připojit 7 hubů pro 127 zařízení. Rozbočovač také určuje hodnoty odběru proudu zařízením (běžně 100 mA až 500 mA).

Rozdělení HUBů:

- kořenový- obsahuje jej každý počítač
- externí- připojuje se k počítači

[7]

6.2 FT232RL- převodník USB- UART

Integrovaný obvod FT232RL je výrobkem firmy FTDI Chip, která byla založena v roce 1990 a její hlavní sídlo se nachází ve skotském Glasgow. Firma se specializuje na konverzi klasických periferií PC na USB. Ucelená řešení firmy FTDI redukují náklady na vývoj a ladění aplikací připojených k PC a zkracují dobu nutnou k uvedení výrobku na trh - díky kombinaci součástek a volně dostupných softwarových driverů pro Windows, Linux a Mac.

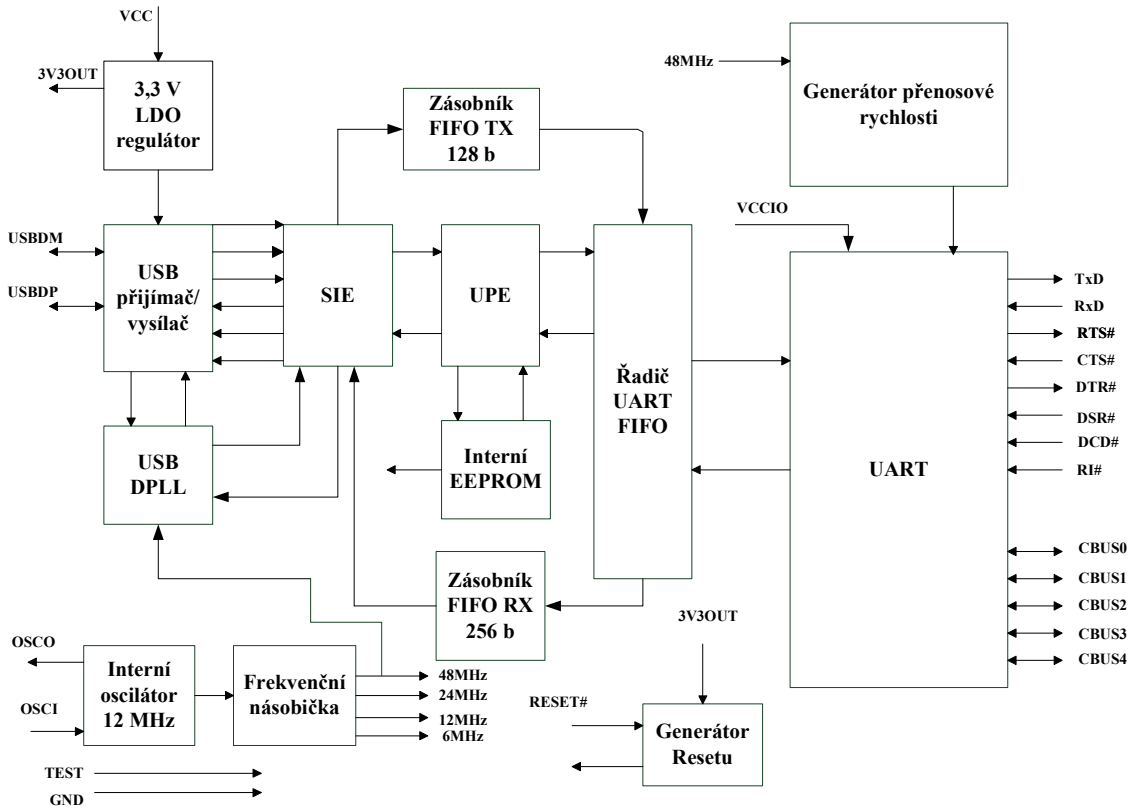
Integrovaný obvod FT232RL tedy slouží jako převodník komunikačního rozhraní USB na asynchronní rozhraní UART. Tento dnes velmi populární USB konvertor je již třetí generací v řadě konvertorů od FTDI. Mezi první generaci těchto obvodů patří FTU232AM, který byl následovně nahrazen velmi oblíbeným FT232MB, tedy druhou generací těchto převodníků.

Velká oblíbenost obvodů druhé generace tkvěla v tom, že kromě nových vlastností poskytovala především částečnou vývodovou kompatibilitu s generací první. Z toho vyplývá, že

z určité části došlo k redukci počtu vnějších součástek, což významným způsobem snížilo náklady na vývoj a výrobu dalších novějších zařízení. Kromě toho se také otevřely možnosti v dalších aplikačních oblastech. Cenová relace USB převodníků druhé generace byla taktéž příznivá, a to do 200 Kč za kus.

Do třetí generace USB- UART konvertorů patří jak už bylo uvedeno na začátku obvod FT232RL. Tento převodník se vyznačuje oproti svému předchůdci dvěmi novými vlastnostmi, a to dokonalejší násobičkou kmitočtu a speciálním ochranným kódem FTDI Chip- ID vpraveným do obvodu již při výrobě, který je čitelný přes USB a slouží k ochraně zákaznických aplikací proti kopírování.

V této bakalářské práci je obvod FT232RL použit pro jednosměrnou sériovou komunikaci s mikroprocesorem vybaveného rozhraním USART, kdy se z osobního počítače přes USB rozhraní posílají data do paměti mikroprocesoru, kde jsou pak dále zpracovávána. Výhodou tohoto obvodu také je, že po konverzi signálu z USB na RS232 pracuje s napětím 5V, a tudíž není nutné používat další napěťový převodník. Na Obr. 12. je znázorněno blokové schéma obvodu FT232RL.



Obr. 12 Vnitřní blokové schéma zapojení obvodu FT232RL

Základní vlastnosti obvodu FT232RL:

- převod USB na UART
- přenosová rychlost od 300 Baud do 1MBaud pro RS232
- přijímací buffer hloubky 256 B a vysílací buffer hloubky 128 B
- integrovaná EEPROM 1024 bit
- přenosový mód USB Bulk
- napájení v rozsahu 3,3 až 5,25V
- podpora X- on / X- off Handshake
- možnost komunikovat pomocí virtuálního portu
- unikátní funkce FTDI Chip- ID

[7], [8], [9]

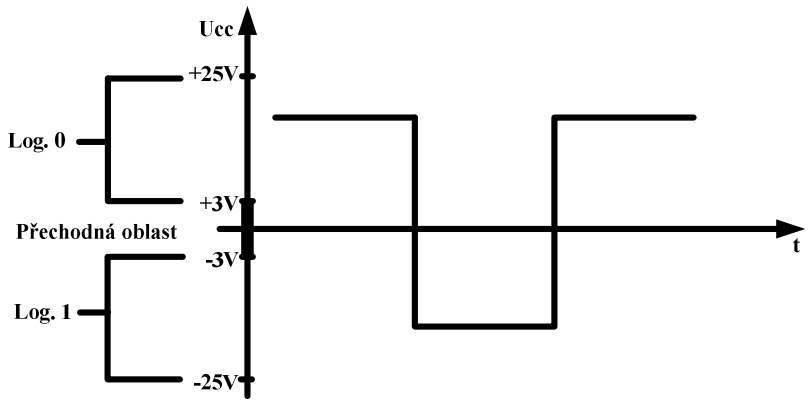
6.3 Komunikační standard RS232

RS232 je rozhraní pro přenos informací vytvořené původně pro komunikaci dvou zařízení do vzdálenosti 20 m. Pro větší odolnost proti rušení je informace po propojovacích vodičích přenášena větším napětím, než je standardních 5 V (pro práci s napájecím napětím 5V je tedy nutné použít napěťové převodníky). Přenos informací probíhá asynchronně, pomocí pevně nastavené přenosové rychlosti.

[10]

6.3.1 Základní parametry RS232

RS 232 používá dvě napěťové úrovně. Logickou 1 a 0. Log. 1 je někdy označována jako marking state nebo také klidový stav, Log. 0 se přezdívá space state. Log. 1 je indikována zápornou úrovní, zatímco logická 0 je přenášena kladnou úrovní výstupních vodičů, jak je znázorněno na Obr. 13.



Obr. 13 Napěťové úrovně přenášeného signálu pomocí RS232

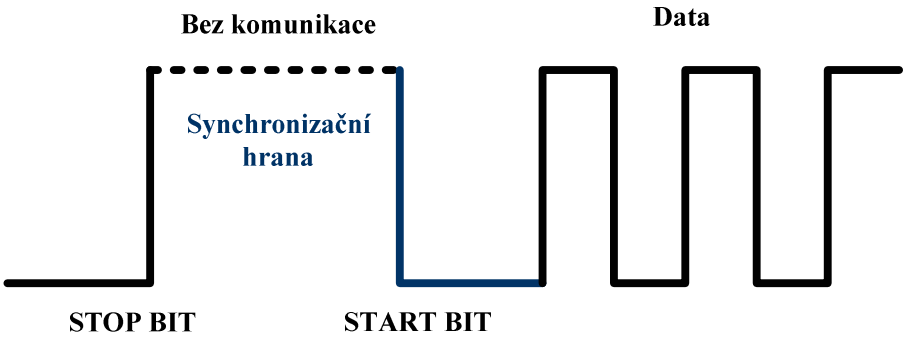
Kompletní přenosová skupina (přenosový rámec) obsahuje přenášená data (7/8 bitová), která jsou doplněná o Start bit, Stop bit a Paritní bit.

Jak již bylo uvedeno, komunikační rozhraní RS232 používá asynchronní přenos dat. Asynchronní přenos dat přenáší data v určitých sekvencích. Data jsou přenášena přesně danou rychlostí a uvozena startovací sekvencí, na kterou se synchronizují všechna přijímací zařízení. K synchronizaci se tedy používá sestupná hrana Start bitu, za kterou již následují posílaná data, jak je patrné z Obr. 14. Všechny strany obsahují vlastní přesný oscilátor, díky kterému odečítají data v přesně definovaných intervalech. Po ukončení sekvence je další příjem opět synchronizován startovní sekvencí.

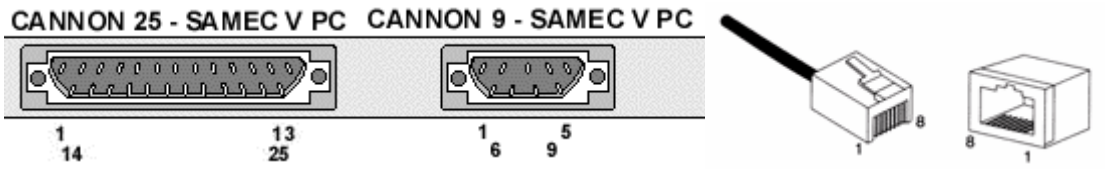
Obecně je asynchronní přenos dat vhodný spíše pro menší objemy dat. Celková rychlost přenášení užitečných dat je až o 20% menší při stejné rychlosti komunikace, vzhledem k nutnosti startovacích a paritních bitů.

Na Obr. 15 jsou vyobrazeny konektory pro sériový přenos a v Tab. 3 jsou uvedeny významy signálů na jednotlivých vývodech.

Při realizaci komunikace mezi PC a Dynamickým zobrazovačem byla použita rychlost přenosu dat 4800 baud. Datový rámec neobsahuje paritní bit, pouze jeden Start bit, Stop bit a data o délce 8 bitů.



Obr. 14 Způsob synchronizace u RS232



Obr. 15 Konektory Canon 25, Canon 9 a RJ45 pro RS232

Název vývodu	Význam vývodu
DCD	Detekce nosného kmitočtu
RxD	Tok dat z modemu do terminálu
TxD	Tok dat z terminálu do modemu
DTR	Oznamuje modemu, že je připraven komunikovat
SGND	Signálová zem
DSR	Oznamuje terminálu, že je připraven komunikovat
RTS	Oznamuje modemu, že komunikační cesta je volná
CTS	Oznamuje terminálu, že komunikační cesta je volná
RI	Indikátor zvonění

Tab. 3 Názvy vývodů a jejich význam v konektorech

[10]

7 Mikroprocesorová část

Prakticky celý řídicí systém Dynamického zobrazovače je založen na mikroprocesoru AVR ATmega16 od firmy Atmel. Tato kapitola tudíž obsahuje základní informace o tomto mikroprocesoru a jeho částech použitých pro řídicí systém.

7.1 Charakteristika mikroprocesoru AVR ATmega16

Postupný vývoj v technologii výroby polovodičových prvků a integrovaných obvodů umožnil vyrábět v současné době univerzální integrované obvody s velmi vysokým stupněm integrace, mezi které patří také mikroprocesory, jejichž charakteristickou vlastností je schopnost realizovat program zapsaný v operační paměti. Během sedmdesátých let prošly mikroprocesory prudkým rozvojem a na současném trhu je velké množství různých mikroprocesorů, které tvoří základ různě složitých mikroprocesorových řídicích systémů pro aplikace v různých odvětvích průmyslu.

Mikroprocesor ATmega16 byl vyvinut a vyroben firmou Atmel Corporation, která se zabývá výrobou polovodičů a integrovaných obvodů. Atmel Corporation má hlavní sídlo ve Spojených státech v Kalifornii. Firma byla založena v roce 1984 a kromě Spojených států má také sídla rozeseta po Evropě (Německo, Francie, Anglie).

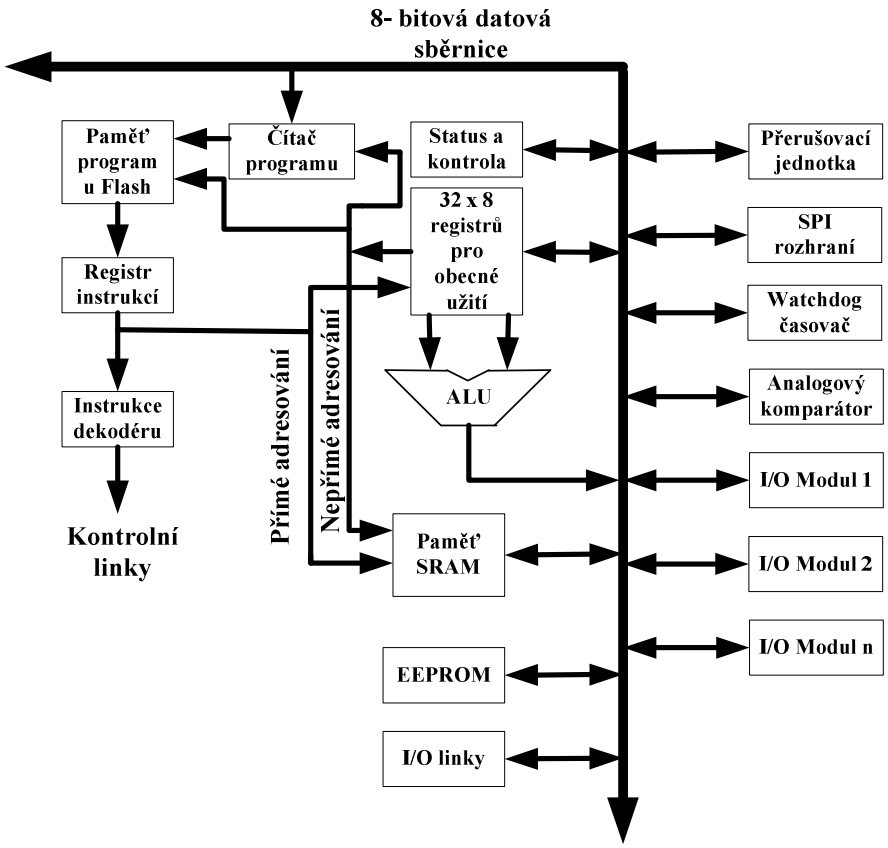
Mezi produkty Atmel patří hlavně mikrokontorléry (včetně klonů 8051, AT91SAM založených na architektuře ARM), jeho vlastní Atmel AVR a AVR32 architektura, rádiové zařízení, EEPROM a Flash paměťové čipy, ASIC, WiMAX a mnoho dalších produktů. Dále pak Atmel podniká ve velkém okruhu aplikačních segmentů včetně: konzumního sektoru, telekomunikace, počítače a počítačové sítě, průmysl, zdravotnictví, automobilový průmysl, letecký a vojenský průmysl. Atmel je také vedoucí firmou na trhu bezpečnostních systémů, hlavně díky čipovým kartám Smart Card a RFID.

ATmega16 je nízkonapěťový 8- bitový CMOS mikrokontrolér založený na rozšířené AVR RISC architektuře. Jádro AVR v sobě spojuje bohatou instrukční sadu s 32 pracovními registry pro všeobecné použití. Všechny tyto pracovní registry jsou přímo napojeny na ALU jednotku, která poskytuje dva další na sobě nezávislé registry, které mají být přístupné v jedné provedené instrukci jednoho hodinového cyklu. I přes velkou rychlost zpracování dat ATmega16 výrazným způsobem optimalizuje spotřebu elektrické energie. Výsledná použitá architektura AVR dosahuje až desetkrát rychlejšího přenosu dat než konvenční mikroprocesory navržené na základě architektury CISC.

Základem architektury AVR je však architektura Harvardská a nikoli Von- Neumanova, čímž jsou mikroprocesory Atmel charakteristické. Harvardská architektura je založena na tom, že veškeré paměti a sběrnice jsou oddělené pro data i program, na rozdíl od architektury Von-Neumanovy, kde jsou tyto prvky společné. Výhoda Harvardské architektury je v tom, že díky použití více pamětí umožňuje paralelní přístup k datům, což výrazným způsobem zlepšuje rychlost zpracovávání všech dat. Architektura mikroprocesoru ATmega16 je na Obr. 16.

Mezi základní vlastnosti mikroprocesoru ATmega16 patří:

- 16 kB programovací paměť FLASH
- 1kB paměť přechodných dat SRAM
- 512 B paměť dat EEPROM
- 2x 8- bitové čítače/ časovače
- 1x 16- bitový čítač/ časovač
- sériové komunikační rozhraní SPI a USART
- rozhraní JTAG
- vnitřní oscilátor
- 4 kanály PWM
- 8x 10- bitový ADC
- frekvence mikroprocesoru 16 MHz



Obr. 16 Harvardská architektura procesorové jednotky ATmega16

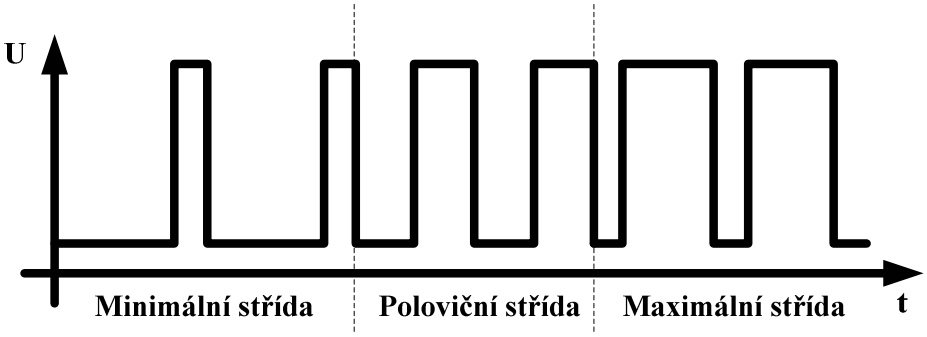
[11], [12]

8 Regulace otáček pomocí PWM

Tato kapitola se zabývá základním popisem PWM modulace, na které je založeno celé řízení otáček stejnosměrného motoru pro promítání dat.

8.1 Princip PWM modulace

Pulsně šířková modulace je druh impulsové modulace, kde nositelem informace je šířka impulsu, tzv. střída- tzn. rozdíl mezi náběhovou a doběhovou hranou, neboli také poměr mezi stavy zapnuto/ vypnuto. Čím je vzorek větší, tím je větší i šířka impulsu. Středy impulsů jsou od sebe vzdáleny rovnoměrně, náběhová a doběhová hrana impulsů je vzhledem ke středu impulsů rovnoměrná. Amplituda všech impulsů je stejně vysoká. Výhodou PWM regulace je to, že těmito rychlými pulsy snižuje celkovou spotřebu elektrické energie, protože je napájecí napětí v podstatě přerušováno. PWM modulace je hojně využívaná ve výkonové elektronice, kde na tomto principu pracují DC/ DC měniče, měniče frekvence, střídače apod. Příklad PWM modulace je na Obr. 17.



Obr. 17 Příklad modulování signálu pomocí PWM modulace

[13]

8.2 Použití PWM modulace pro řízení otáček stejnosměrného motoru

Obecně se každý stejnosměrný motor skládá ze statoru se sběracím ústrojím a z rotoru(kotvy) s komutátorem. Činnost stejnosměrných motorů se zakládá na silovém účinku magnetického pole na vodič, jímž prochází proud. Do vinutí kotvy se přivádí proud pomocí kartáčů. Po připojení napájecího napětí na kotvu protéká vinutím kotvy proud, čímž se na ní indukuje napětí a vzniká magnetické pole. Vzájemným působením magnetického pole statoru a kotvy vznikne síla, kterou se vyvine točivý moment motoru. Komutátor pak zajišťuje změnu toku proudu (přepólování), aby se zachoval směr otáčení. Síla vzniklá vzájemným působením magnetických polí se vypočte následovně:

$$F = B \cdot I \cdot l$$

Indukované napětí na rotoru motoru odpovídá:

$$U_i = \phi \cdot n$$

Celkové napájecí napětí je pak dáno vztahem:

$$U = U_i + R_v \cdot I$$

Z uvedeného vztahu pro indukované napětí vyplývá, že otáčky motoru jsou tím větší, čím menší je magnetický tok, a tudíž i proud tekoucí vinutím. Ovšem proud se mění podle zátěže, kterou musí motor překonat a protékat vinutím bude jen tehdy, pokud je připojeno napájecí napětí.

Použité PWM řízení otáček tedy podle nastavené střídy určuje, jak dlouho bude protékat proud vinutím kotvy. Při nejnižší střídě to bude tak krátký okamžik, že se motor neroztočí vůbec. Ale při jejím zvyšování se tento interval prodlužuje a po překonání původního odporu se motor již roztočí. Dalším zvětšováním střídy jsou impulsy přerušovány velmi rychle, což má za důsledek zvyšování otáček motoru vlivem setrvačnosti. Toto platí i obráceně. Po překonání počátečního odporu se motor snižováním střídy dostane do otáček nižších.

[14]

8.2.1 Obvod NE555

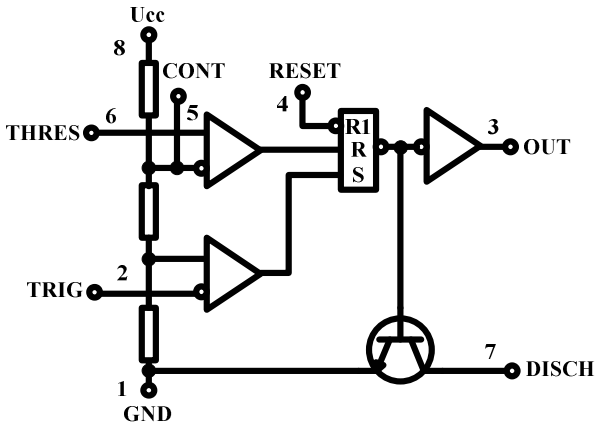
Základním prvkem, pomocí kterého byla v této bakalářské práci realizována modulace PWM pro regulaci ptáček stejnosměrného motoru je obvod NE555, což je integrovaný obvod používaný nejčastěji jako časovač nebo generátor různých pravoúhlých signálů.

Obvod obsahuje dva komparátory a jeden klopný obvod na výstupu. Komparační úrovně jsou odvozeny z děliče napětí sestávajícího ze tří 5- ti kΩ rezistorů. Přestože byl původně navržen pro časovací účely, je tak univerzální, že se dá použít na obrovské množství jiných aplikací jako např. generátory zvuků, měřiče kmitočtů, převodníků, atd.

Napájecí napětí, přivedené na pin 8 (*VCC*), se na rezistorech R1 – R3 rozdělí na 1/3 a 2/3 své hodnoty, čímž se nastaví komparační úrovně dvou vstupních operačních zesilovačů, které jsou zapojeny jako komparátory. Vstupem 5 (*CTRL*) lze tyto úrovně měnit. Výstupy obou komparátorů jsou spojeny se vstupy klopného obvodu RS. Tento KO má ještě jeden, nulovací vstup (pin č.4, *RESET*), kterým je možné překlopit jej do stavu logické nuly nezávisle na jeho vstupech. Invertující výstup KO je přiveden na invertor a ten pak na pin číslo 3 (*OUT*). Na tento invertující výstup KO je také připojena báze vybíjecího tranzistoru. Pokud je KO RS překlopen do stavu logické nuly, je na bázi tohoto tranzistoru log. jednička. Ta způsobí otevření tranzistoru, čímž se jeho kolektor (pin č.7, *DIS*) spojí se zemí. Na Obr. 18 je znázorněno vnitřní zapojení obvodu NE555 a v Tab. 4 jsou názvy a významy jednotlivých vývodů.

Č. pinu	Označení	Popis
1	GND	uzemění obvodu, 0V
2	TRIG	spouštění, vstup druhého (zapínacího) komparátoru
3	OUT	výstup obvodu
4	RESET	nulovací vstup, umožňuje nulování KO nezávisle na vstupech
5	CTRL	řídicí napětí, ovlivňuje překlápění komparátorů
6	THR	práh, vstup prvního (vypínacího) komparátoru
7	DIS	vybíjení, kolektor vybíjecího tranzistoru
8	VCC	kladné, napájecí napětí v rozsahu 4,5 V až 15 V

Tab. 4 Popis jednotlivých vývodů obvodu NE555



Obr. 18 Vnitřní zapojení obvodu NE555

[15]

9 Zobrazovací část a její princip

Nejdůležitější složkou celého systému Dynamického zobrazovače je jeho promítací část. Bez ní by celá aplikace neměla smysl. Proto je v této kapitole popsán její teoretický princip, způsob řešení a podmínky pro správnou funkci.

9.1 Princip zobrazování

Základ celého promítacího systému je v jednom sloupci 16- ti LED, které jsou rozmístěny těsně vedle sebe. Každá z jednotlivých LED reprezentuje jeden samostatný vykreslovací bod promítaného textu nebo obrázku. Smysl zobrazování, který je zde použit spočívá v tom, že celý obrázek je digitálně rozdělen (vzorkován) na jednotlivé vzorky pomocí uživatelského editačního softwaru v PC. Obrazec je vykreslován tímto sloupcem LED diod postupně celý pomocí vzorků, ze kterých se skládá.

K tomu, aby tento způsob zobrazování měl smysl, tak se musí celý obrazec vykreslovat v rychlém sledu za sebou, aby nebylo lidské oko schopné rozlišit jak po sobě následují. Tento rychlý sled stejných na sebe navazujících obrazců určuje velikost otáček motoru, který rotuje s celým vykreslovacím sloupcem LED. Pomyslná hodnota otáček motoru pro správné vykreslování je kolem 1500 až 2000 ot./ min., což odpovídá době otáčky od 40 do 30 ms. Čím rychleji se tedy motor točí, tím lepší a kvalitnější vykreslování je. Pokud klesne hranice otáček pod 1500 za minutu, tak je v tomto případě lidský zrak do jisté míry schopen rozeznat jednotlivé za sebou jdoucí obrazce, což se projeví např. tím, že celý obrazec nebo text bliká. Kromě toho se díky pohybu sloupce LED vytvoří v podstatě jakýsi rastr nebo obrazovka.

Dále je nutné zajistit, aby se jednotlivé vzorky obrazce vykreslovaly po určitou časovou konstantu t_s, která musí splňovat Shanonův a Kotělníkův teorém o vzorkování signálu s diskrétním časem. Tato časová konstanta je odvozena z otáček motoru při daném rozlišení 16 x 64 musí být tolikrát menší než je celá doba jedné otáčky, protože vykreslování zajišťuje pouze

jeden sloupec s LED. V Tab. 5 jsou uvedeny příklady hodnot otáček motoru s nutnými hodnotami pro časovou konstantu t_s pro vykreslení jednoho vzorku (sloupce) obrazce vzhledem k rozlišení. Celý princip je dále patrný z Obr. 19.

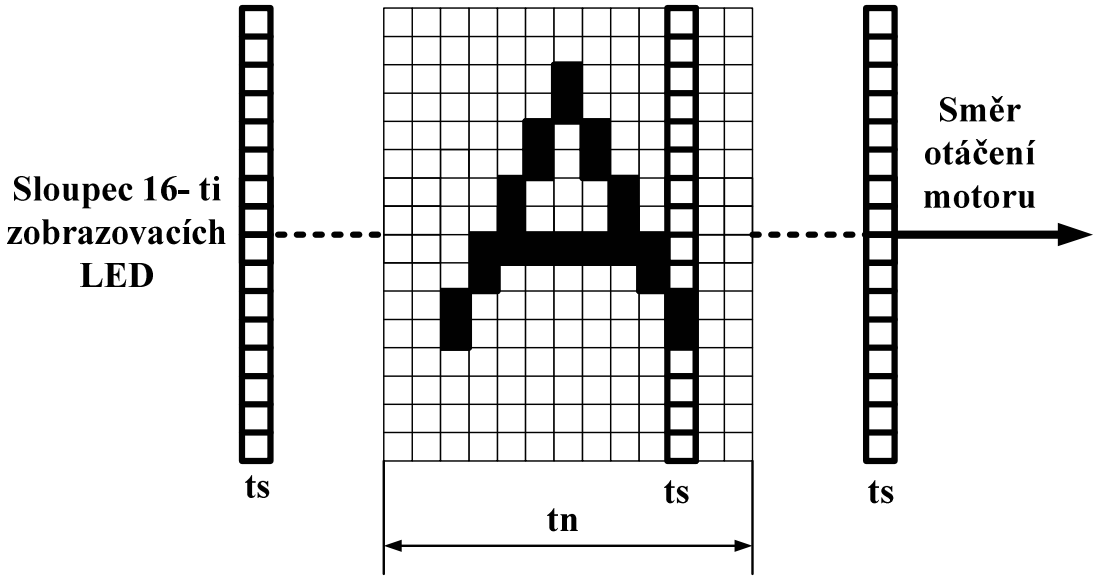
n	t_n	t_s
[min.]	[ms]	[μs]
500	120	1875
800	75	1171
1000	60	937,5
1500	40	625
2000	30	468,75
2300	26	406,25
2500	24	375
3000	20	312,5
3500	17	265,625

Tab. 5 Příklady ručně vypočtených hodnot t_s pro různé hodnoty otáček

Zde je uveden příklad výpočtu pro jednotlivé hodnoty v tabulce:

$$n = 2000 ot./\min. \Rightarrow \frac{2000}{60} = 33{,}34 ot/s \Rightarrow t_n = \frac{1}{33{,}34} = \underline{30 ms}$$

$$t_s = \frac{t_n}{64} = \frac{30 \cdot 10^{-3}}{64} = \underline{\underline{468{,}75 \mu s}}$$



Obr. 19 Princip zobrazovací části

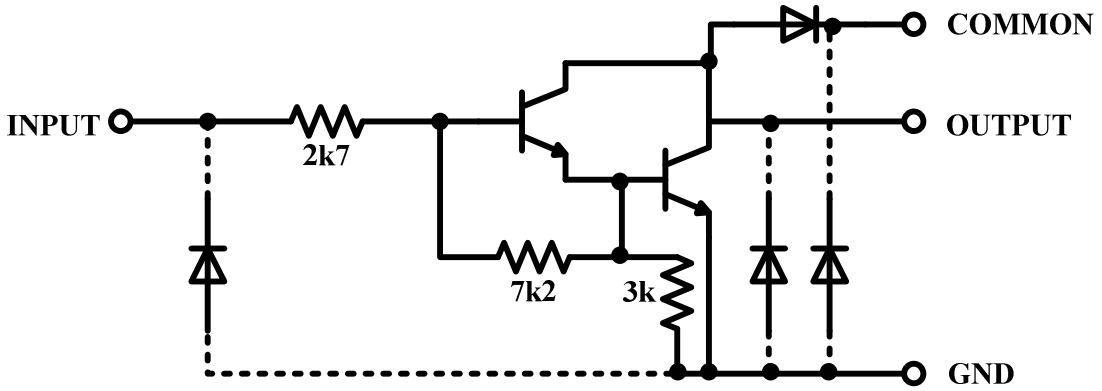
9.1.1 Spínání zobrazovacích LED

Z MCU se vysílají digitální signály, které rozsvěcují nebo zhasínají příslušné zobrazovací body realizované pomocí vysoce svítivých LED. Tyto LED diody mají poměrně značnou spotřebu elektrické energie, a kdyby byly zapojeny přímo na výstupy MCU, tak by to mohlo vést v extrémním případě(např. při všech 16- ti rozsvěcených LED) až k jeho přehřátí a následné destrukci nadměrnou proudovou zátěží.

Proto byl použit princip spínání tranzistorem každé samostatné LED zvlášť. Při použití tranzistoru jako spínače, je proud tekoucí do jeho báze pro otevření zanedbatelný, takže MCU nijak zvlášť proudově nezatíží. Proud tekoucí LED je odebírán přímo ze zdroje a neprotéká vůbec mikroprocesorem. Navíc tranzistor v otevřeném stavu má velmi malý odpor, takže se dá také říci, že ztráty na něm jsou minimální.

Navíc bylo použito Darlingtonova zapojení bipolárních tranzistorů, které ještě zvyšuje proudové zesílení, a proto je proud do báze tekoucí přes MCU potřebný k otevření prvního tranzistoru skutečně velmi malý, čímž se výrazně sníží energetické nároky celého systému řídicí elektroniky a zvýší se tím i jeho spolehlivost. Pro dosažení ještě větší integrace systému byla použita tranzistorová pole s Darlingtonovým zapojením tranzistorů, které je vyobrazeno na Obr. 20.

Darlingtonův způsob zapojení tranzistorů spočívá v tom, že první tranzistor svým kolektorovým proudem otevírá tranzistor druhý.



Obr. 20 Vnitřní zapojení tranzistorového pole s Darlingtonovým zapojením tranzistorů [16]

10 Konstrukční návrh dynamického zobrazovače

K tomu, aby celý systém fungoval úspěšně je nezbytné, aby celá zobrazovací část s LED rotovala. Pro tento účel byl zvolen stejnosměrný motor. Z toho ovšem ještě vyplývá, že kromě použití stejnosměrného motoru k roztáčení celé elektroniky je třeba ještě navrhnout mechanický systém napájení celého řídicího systému tak, aby i při vyšších otáčkách motoru byl schopen dostatečně napájet celou aplikaci. Konečný návrh konstrukce je zobrazen na Obr. 21. Detailnější obrázky na celé zařízení je pak obsaženo v příloze.

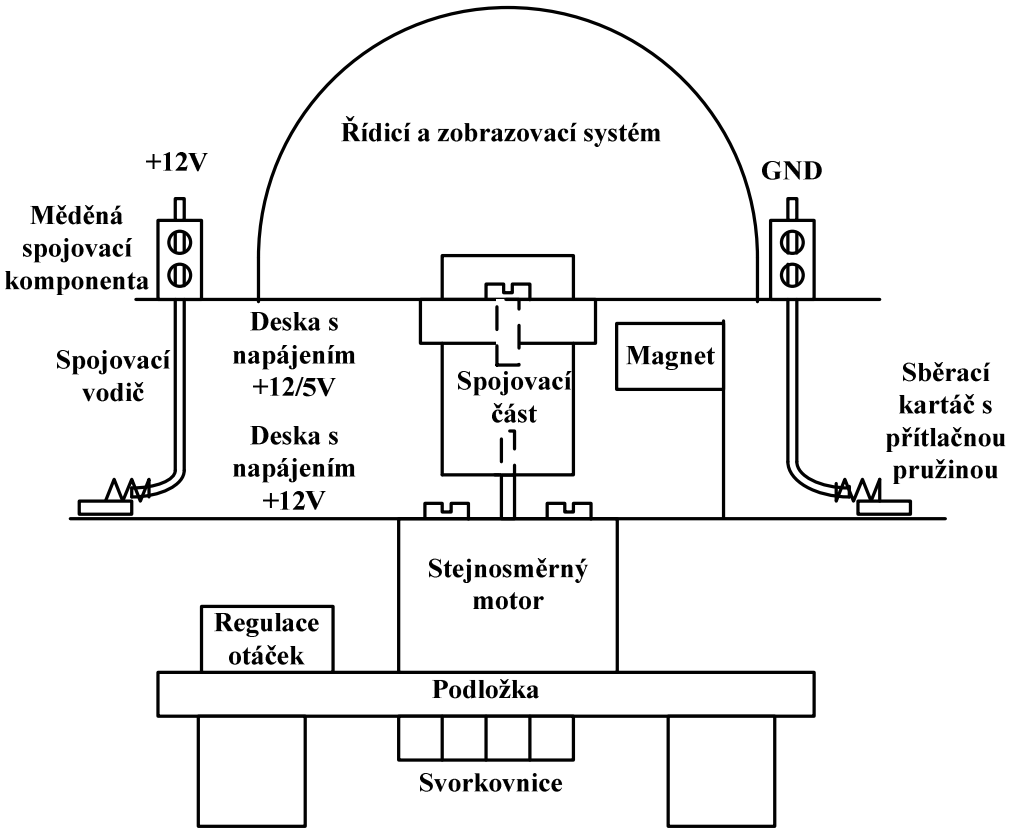
Motor je uchycen v dřevěné podložce tak, že otvor pro jeho umístění byl vyvrtán velmi těsně, čímž byl motor po vpravení kolmo do podložky pevně uchycen a stabilizován. Zbylé vůle díky vzniklé malé mezeře byly vyplněny sekundovým lepidlem, a tím byl motor spolehlivě uchycen do podložky tak, že se při své činnosti nemůže uvolnit. Na podložce je rovněž přichycena DPS s regulátorem otáček motoru pro snadnou obsluhu celého zařízení a svorkovnice, která zajišťuje přívod napájení jak pro motor, tak pro celou řídicí elektroniku..

Připevnění DPS řídicího systému k motoru je realizováno pomocí spojovací části, která byla pro tento účel vytvořena. Ve spodní části je vyvrtán otvor o stejném průměru jako má hřídel motoru, do něhož byla tato hřídel vsazena. Tím bylo dosaženo velké těsnosti, která po pečlivém vsazení velmi dobře drží na hřídeli motoru. K DPS je spojovací část připevněna pomocí šroubu M4. Do spojovací části byl proto vyřezán příslušný závit, protože samo řezný šroub by po čase mohl jím vyřezané závity poškodit a celý systém by se mohl při provozu rozpadnout. Celá spojovací část byla zhotovena z tvrzeného plastu aby se zajistila dostatečná houževnatost a zároveň aby co nejméně zatěžovala motor.

Základem napájení řídicí elektroniky je DPS připevněna do příslušných otvorů motoru pomocí šroubů M3. Na této DPS jsou vyleptány vodivé cesty ve tvaru kruhu, na které jsou přivedeny jednotlivé potenciály napájecího napětí. Po těchto dráhách pak při chodu zařízení jezdí sběrací kartáče podobně jako na autodráze. Tyto sběrače jsou pak připevněny pájením ke přítlačné pružině, která svým tlakem proti podložce zajišťuje dostatečný kontakt kartáče s vodivou cestou. Z druhé strany pružiny je připájen zespodu zahnutý (pro co nejmenší vůli při jízdě po vodivých cestách) vodič, který je již spojen s DPS rotující elektroniky. Toto připojení vodičů je realizováno pomocí mosazných komponent svorkovnic s otvory a závity. Pomocí šroubů jsou pak tyto vodiče napevno přichyceny, neboť celá mosazná komponenta je ještě připájena k DPS.

K DPS s napájením je ještě připevněn stojan s magnetem, který je zapotřebí ke snímání otáček pomocí Hallova senzoru.

Hlavní DPS s řídicí a zobrazovací částí byla opracována do tvaru půlkruhu, což pak vytváří efekt prostorového zakřivení, který je vhodný pro vykreslení např. polokoule. Tato DPS je pak ke druhé DPS připevněna pomocí pinové lišty pájením.



Obr. 21 Konstrukční návrh aplikace

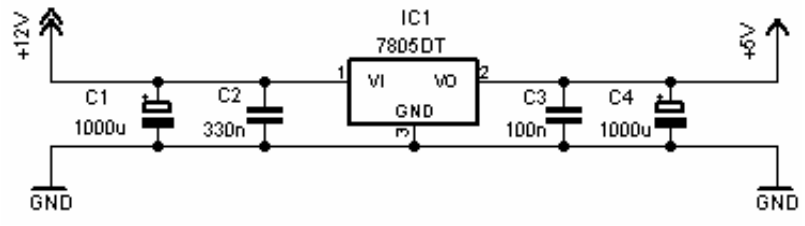
11 Návrh elektroniky řídicího systému

V této kapitole je obsažen popis celého elektronického systému a jeho jednotlivých částí z hlediska principu i z hlediska funkce elektrického zapojení. Všechny DPS byly vytvořeny v programu EAGLE a jsou společně se schématy zapojení k dispozici v příloze 3.

11.1 Schéma zapojení napájecího zdroje +12V/ +5V

Na Obr. 22 je schéma zapojení napájecího zdroje pro řídicí elektroniku Dynamického zobrazovače. Pro jeho realizaci byl použit integrovaný stabilizátor napětí 7805 v SMD pouzdře. Tento stabilizátor napětí potřebuje ke své správné činnosti vstupní napětí minimálně o 2V vyšší než je jeho výstupní stabilizovaná hodnota, díky úbytku napětí, který na stabilizátoru vznikne. V tomto případě bylo zapotřebí výstupní napětí 5V, takže minimální hodnota vstupního napětí je 7V. Podle katalogových listů tohoto obvodu lze jako vstupní napětí použít až 35V a maximální proud, který může být odebírán je 1A, což jsou dostatečné hodnoty. Maximální odebíraný proud by se totiž měl podle spotřeby zobrazovací části pohybovat okolo 320 mA. Odběr ostatních obvodů je zanedbatelný. Vstupní napětí bylo použito o velikosti +12V. Vstupní kondenzátor C1 a výstupní kondenzátor C2 byly zvoleny 1000 μF s ohledem na způsob napájení pomocí sběrných kartáčů. Při rychlejším pohybu kartáčů se může stát, že některý z nich vyjede

ze své dráhy a napájecí napětí by bylo přerušeno. Poměrně vysokou hodnotou těchto dvou kondenzátorů na vstupu i výstupu bylo zajištěno, že i při krátkodobějším výpadku napájení zůstane elektronika napájena. Odrušovací kondenzátory pak byly zvoleny dle katalogu na hodnoty C2= 330nF a C3= 100nF.



Obr. 22 Schéma zapojení napájení pro řídicí elektroniku Dynamického zobrazovače [17]

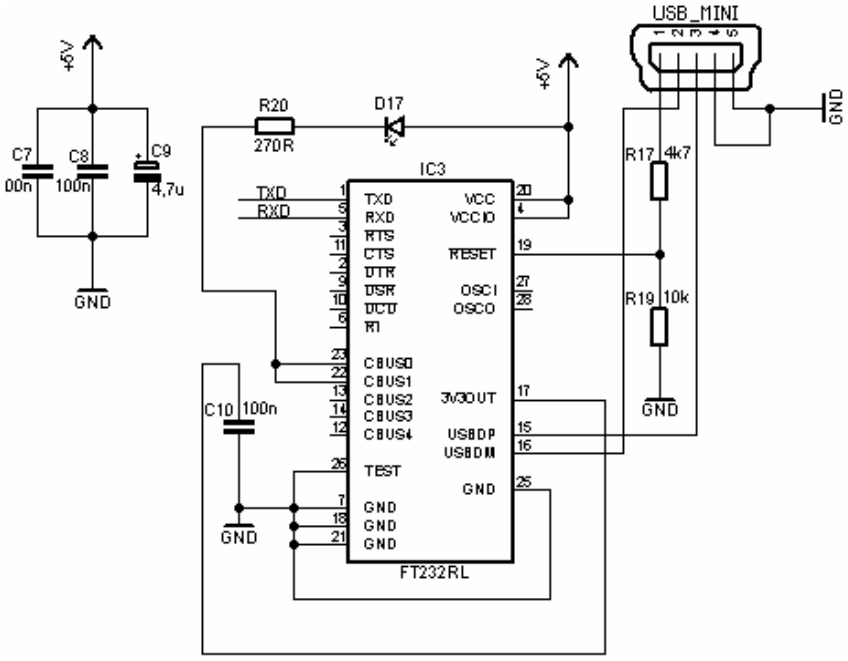
11.2 Schéma zapojení převodníku FT232RL

Na Obr. 23 je schéma zapojení pro převod dat přenášených pomocí sběrnice USB na sběrnici RS232. Samotná realizace je provedena pomocí obvodu FT232TL.

Napájecí vývod konektoru USB je použit ke kontrole zařízení. Je- li USB HUB napájen přes vnitřní pull- up rezistor 1,5 kΩ, tak se na pinu převodníku USBDP objeví napětí +3,3 V (odporový dělič z rezistorů 4k7 a 10k) a určí se, že se jedná o USB zařízení typu Full Speed. Pokud je ovšem USB HUB vypnutý, tak přes pull- up rezistor neprotéká žádný proud a na odporovém děliči je nízká úroveň napětí a obvod vykoná Reset.

Napájení je zde použito z aplikace nikoliv přímo z USB konektoru. Data z osobního počítače proudí přes USB konektor na vstupy USBDP (data z vývodu D+) a USBDM (data z vývodu D-). Po samotném převodu jsou pak dále posílána výstupy TxD a RxD do mikroprocesoru.

Mikroprocesor disponuje rozhraním USART, a proto jsou signály TxD a RxD připojeny k příslušným vývodům mikroprocesoru se společnou GND. LED 17 s předřadným rezistorem indikuje příjem dat.



Obr. 23 Schéma zapojení převodníku FT232RL pro posílání dat při napájení z aplikace [9]

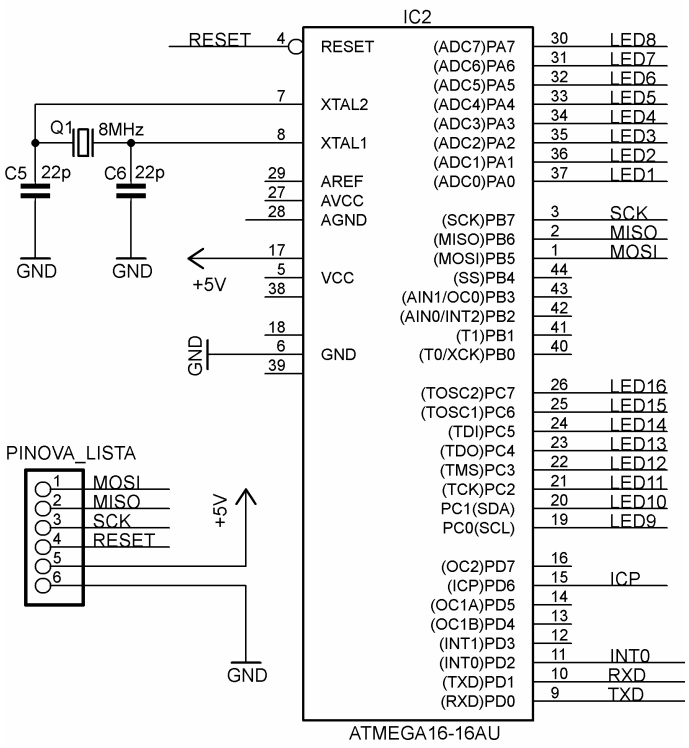
11.3 Schéma zapojení mikroprocesoru

Mikroprocesor je hlavní součástí celého řídicího systému. Jsou k němu připojeny veškeré signály které je třeba zpracovávat a zároveň jsou jím vysílány signály, které následně vykreslují zadaný obrazec. Na Obr. 24 je znázorněno schéma zapojení MCU.

K tomu, aby mohl mikroprocesor vůbec fungovat je zapotřebí připojit ho k napájecímu napětí, které je v tomto případě +5V ze stabilizátoru. Dále jsou připojeny všechny příslušné programovací piny (MISO, MOSI ,SCK, RESET), aby bylo možné mikroprocesor naprogramovat. Frekvenci konání instrukcí pak zajišťuje rezonátor složený s 8MHz krystalu a dvou kondenzátorů C5 a C6 oba o hodnotě kapacity 22pF.

K přijímači editovaných dat RxD jednotky USART je připojen vysílací pin konvertoru FT232RL TxD (připojen je i pin RxD, takže komunikace by mohla být i obousměrná). Informace o aktuálním stavu otáček z Hallovy sondy jsou odesílány na vývod ICP a zahájení přenosu dat pomocí tlačítka je indikováno v externím přerušení INT0.

K portu A i k portu C je pak připojeno všech 16 zobrazovacích LED, kdy oba tyto porty jsou nastaveny jako výstupní.



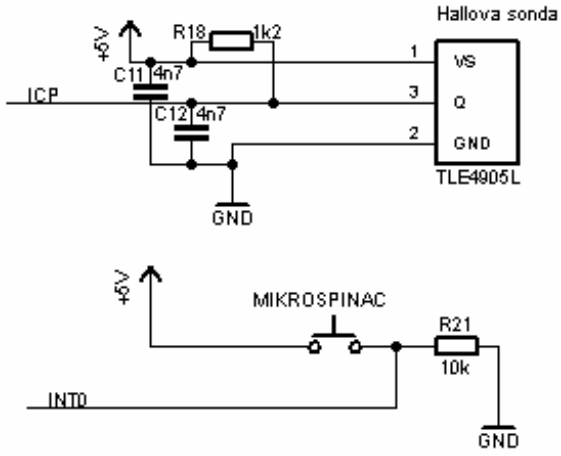
Obr. 24 Schéma zapojení MCU

[12]

11.4 Zapojení Hallova senzoru a mikrospínače

Na Obr. 25 nahoře je zapojena Hallova sonda, která posílá MCU informace o aktuálním stavu otáček stejnosměrného motoru. Toto zapojení bylo použito z příslušného katalogu senzoru TLE4905L. Kondenzátory C11 a C12 nastavují svou dobou nabíjení dobu náběhu a doběhu výstupního signálu, který se objeví na předepsané zátěži rezistoru R18 1k2. Výstupní impulsy se odesílají do mikroprocesoru na ICP.

Mikrospínač je zapojen k +5V a přes rezistor R21 ke GND. Takže při stisknutí se na vývodu INT0 mikroprocesoru objeví úbytek napětí na R21, což odpovídá log. 1. Výhodou je, že při stisku tlačítka je odebírán proud pouze při sepnutí, což trvá zanedbatelnou chvíli. Hodnota rezistoru R21 byla zvolena na 10k pro nízký odběr proudu, což se v praxi osvědčilo.



Obr. 25 Zapojení Hallovy sondy a mikrospínače

[6]

11.5 Zapojení zobrazovacích LED se spínacími tranzistory

Jak již bylo uvedeno, zobrazovací LED jsou na základě výstupních signálů z MCU spínány pomocí Darlingtonova zapojení bipolárních tranzistorů. Takže při úrovni log. 1 z výstupu MCU se otevřou oba tranzistory a LED proteče proud, dokud se úroveň nepřepne na log. 0. Tím se dosáhne minimálního proudového zatížení mikroprocesoru a tranzistory jako takové mají při saturaci velmi malý dynamický odpor, takže případné ztráty na tranzistoru jsou rovněž velmi malé.

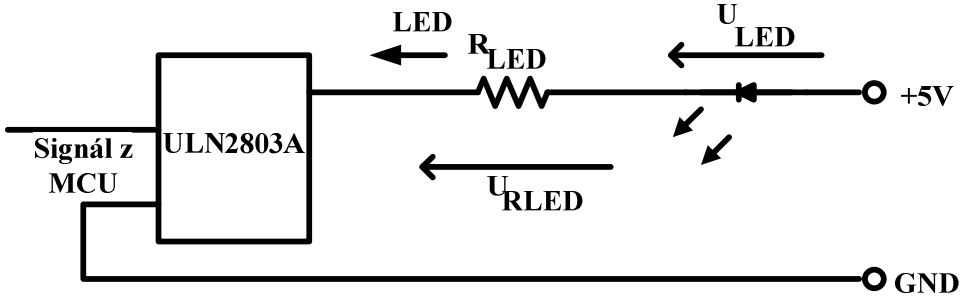
Použité LED byly zvoleny vysoce svítivé. Pro dosažení co největšího svitu byl proud protékající přes každou LED nastaven předřadným rezistorem na maximální přípustnou hodnotu, což bylo v tomto případě 20mA. Úbytek napětí na LED činí 3V a napájecí napětí je 5V. Obr. 26 demonstruje zapojení, ze kterého bylo vycházeno při výpočtu předřadného rezistoru.

Na Obr. 27 je již celé zapojení zobrazovacích LED spínaných Darlingtonovým zapojením tranzistorů v tranzistorovém poli ULN2803.

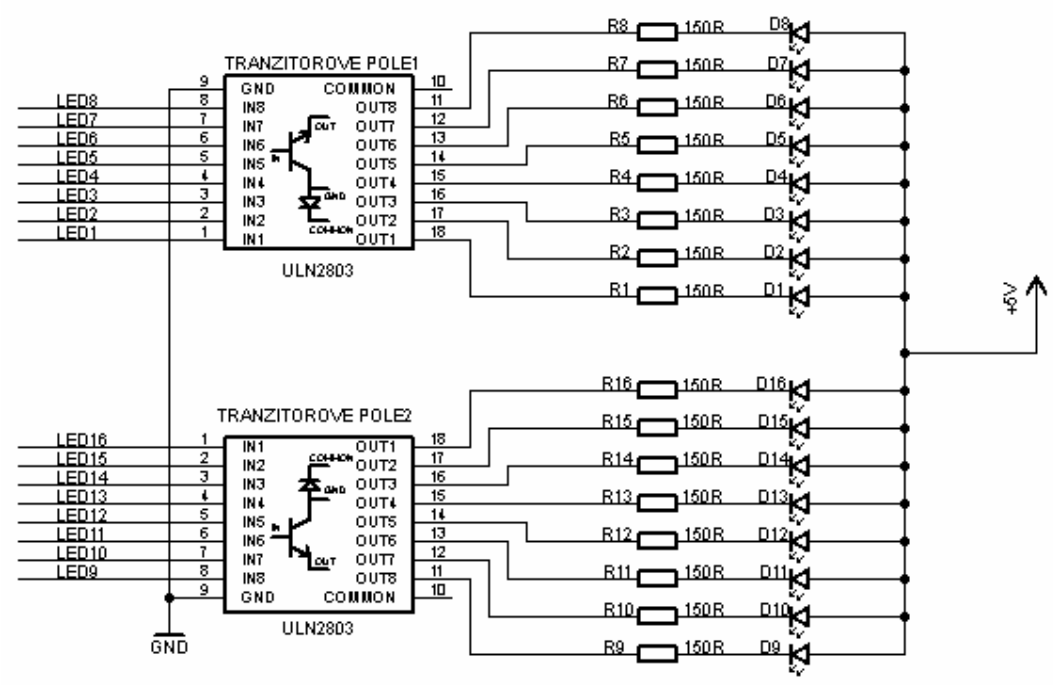
Na základě výše uvedených hodnot byl výpočet odporu předřadného rezistoru proveden následovně:

$$U_{CC} = 5V$$
$$I_{LED} = 20mA$$
$$\underline{U_{LED} = 3V}$$

$$U_{CC} = U_{RLED} + U_{LED} \Rightarrow U_{RLED} = U_{CC} - U_{LED} = 5 - 3 = \underline{2V}$$
$$R_{LED} = \frac{U_{RLED}}{I_{LED}} = \frac{3}{20 \cdot 10^{-3}} = \underline{\underline{150\Omega}}$$



Obr. 26 Zapojení LED k tranzistorovému poli pro výpočet předřadného rezistoru



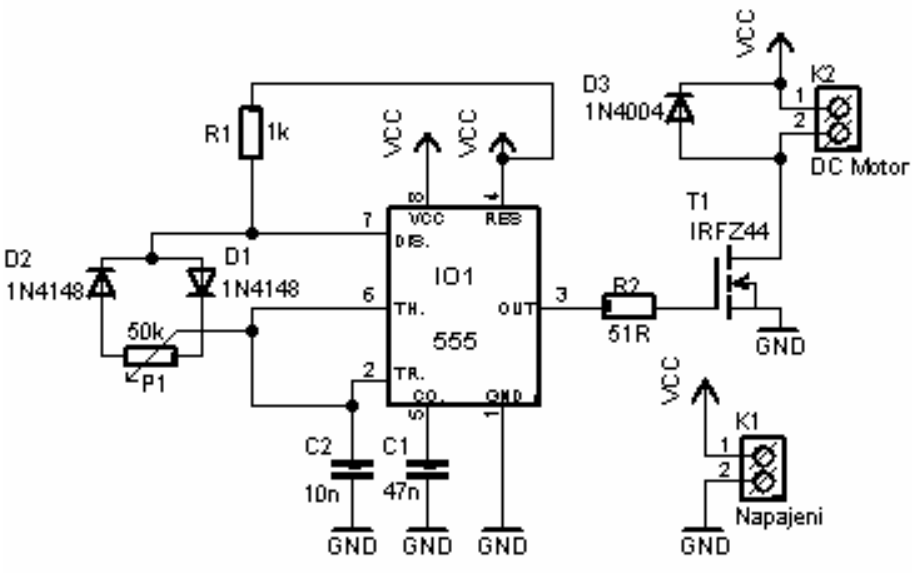
Obr. 27 Schéma zapojení zobrazovací části s 16- ti LED

11.6 Schéma zapojení řízení otáček motoru pomocí PWM

Na Obr. 28 je znázorněno schéma zapojení pro regulaci otáček motoru s využitím PWM modulace.

Základem tohoto zapojení je astabilní klopný obvod (multivibrátor). Multivibrátor je v podstatě generátor impulsů, což je základ pro vytvoření modulace PWM. V tomto případě byl pro realizaci multivibrátoru použit obvod NE555. Po připojení napájecího napětí se začne nabíjet kondenzátor C2, protože je vybíjecí tranzistor uzavřený. Po nabití kondenzátoru na hodnotu dvou třetin napájecího napětí VCC dojde k překlopení komparátoru (vstup č. 6), čímž se resetuje klopný obvod RS a na výstupu se objeví log.0. Zároveň se ale otevře vybíjecí tranzistor a kondenzátor C2 se začne vybíjet. Až se vybije na hodnotu menší než jedna třetina VCC dojde k překlopení druhého komparátoru, který nastaví klopný obvod RS a na výstupu se

tím pádem objeví log.1 a vybíjecí tranzistor je tímto uzavřen. Následně dochází k opětovnému nabíjení kondenzátoru a celý proces se neustále opakuje. Doba nabíjení a vybíjení kondenzátoru pak určuje dobu trvání jednotlivých impulsů a mezer mezi nimi.. Pomocí dvou diod D1 a D2 a potenciometrem P1 byla rozdělena nabíjecí a vybíjecí cesta, čímž lze nastavit prakticky libovolnou střídu. Takovéto impulsy o nastavené střídě pomocí potenciometru jsou posílány na bázi spínacího tranzistoru, který je těmito pulsy spínán. Spínáním tranzistoru dle nastavené střídy dochází v podstatě ke přerušovanému připojování a odpojování napájecího napětí na stejnosměrný motor a vlivem setrvačnosti se pak tento efekt projeví jako regulace jeho otáček. Nejvýhodnější se jeví použití unipolárního tranzistoru, díky svému velmi malému dynamickému odporu v otevřeném stavu, což výrazně sníží ztrátový výkon na samotném tranzistoru. Dioda D3 je ochranná proti spínání indukční zátěže (motoru). Tranzistor byl pak ještě vybaven chladičem pro lepší odvod tepla při roztáčení motoru.



Obr. 28 Zapojení regulátoru otáček

[18], [19]

12 Popis softwaru pro PC a MCU

Řídicí systém pro aplikaci Dynamického zobrazovače není realizován čistě jen pomocí hardwaru a elektronického systému, ale z velké části i softwarem jak pro MCU tak pro PC, neboť bez použití správného softwaru by mikroprocesor nemohl pracovat a jakákoli komunikace z PC by byla vyloučená. Proto byl navrhnut uživatelský software a software pro mikroprocesor, kterými se tato kapitola zabývá.

12.1 Editační uživatelský software

Na této softwarové aplikaci jsem spolupracoval s Richardem Adamovským.

Tento program byl vytvořen v prostředí Microsoft Visual Studio 2008 v programovacím jazyce C#. Programovací jazyk C# je poměrně výkonný a jednoduchý jazyk především pro vývoj aplikací v prostředí .NET Framework v rámci Visual Studia, které vytváří jednoduché a efektivní grafické programové rozhraní.

Význam této programové aplikace spočívá především v tom, že díky němu je celá obsluha a funkčnost Dynamického zobrazovače mnohem lepší. Tento program totiž vložený obrázek nebo text, který v něm lze vytvořit převede na binární kód, který je pak využit v mikroprocesoru. Tento binární kód lze rovněž prostřednictvím tohoto programu poslat do Dynamického zobrazovače, tudíž tvoří uživatelské rozhraní.

Bez existence této aplikace by nebyl provoz Dynamického zobrazovače tak plynulý, a k tomu, aby docházelo k vykreslování různých obrazců nebo textů by bylo zapotřebí neustále měnit řídicí program v MCU podle toho, co by chtěl uživatel zobrazovat.

12.1.1 Popis funkce editačního softwaru

Na Obr. 29 je vývojový diagram celé aplikace, který demonstruje jeho funkci. Pro větší přehlednost celého diagramu slouží následující slovní popis.

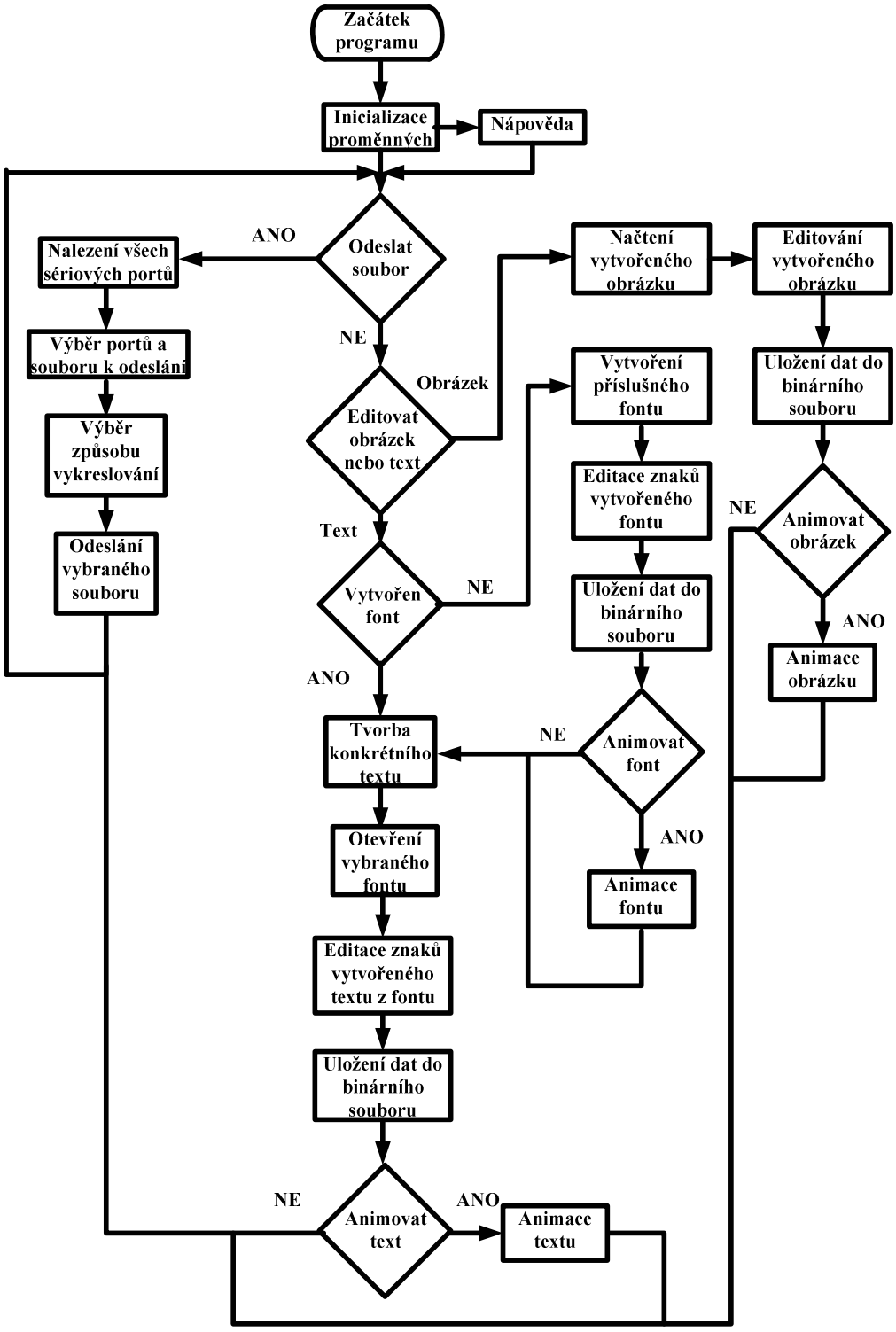
Po spuštění celé aplikace má uživatel na výběr, co chce podniknout. Pokud ještě software nezná, tak se může podívat do připravené nápovědy, která mu objasní, jakým způsobem s programem zacházet. Po přečtení nápovědy již uživatel může s programem začít pracovat a to tak, že si nejprve vybere co chce dělat. Pokud ještě nemá vytvořený soubor k odeslání nebo chce vytvořit soubor jiný, musí se rozhodnout, jestli chce editovat obrázek nebo text.

Pokud si vybere editaci textu, musí mít nejprve vytvořen příslušný font písma. Pokud jej vytvořen nemá, tak si jej musí vytvořit tak, že si vybere v záložce Font jeho typ a velikost a stiskne tlačítko pro vytvoření fontu. Po této akci se příslušné znaky editují a uloží do uživatelem pojmenovaného binárního souboru. Po vytvoření fontu už pak stačí jen tento font tlačítkem otevřít a napsat text. Po následném stisku tlačítka pro vytvoření textu bude tento text editován na základě vytvořeného fontu a uložen opět do binárního souboru.

Tento soubor již pak může uživatel odeslat do Dynamického zobrazovače. Po připojení USB kabelu se uživateli zobrazí všechny dostupné sériové porty na PC. Po vybrání správného portu si uživatel vybere soubor k odeslání a tlačítkem ho odešle do aplikace. Tento postup platí pro každý editační soubor nehledě na text nebo obrázek. Ještě před odesláním souboru si může uživatel zvolit mezi stylem vykreslování (statické nebo pohyblivé vykreslování).

Při výběru editování obrázku je postup obdobný s tím rozdílem, že je nutné již předem vytvořený černobílý obrázek v přesně stanoveném rozlišení (16 x 64) načíst pomocí tlačítka. Po načtení lze pak již stisknutím tlačítka pro editaci obrázek editovat a uložit editační data do binárního souboru.

Součástí programu je i animace která slouží jako náhled editovaných znaků nebo obrazců. Animovat lze jak fonty, vytvořené texty, tak obrázky.



Obr. 29 Vývojový diagram Editačního uživatelského rozhraní

12.1.2 Ukázky nejdůležitějšího zdrojového kódu uživatelského rozhraní

V této část jsou uvedeny jednotlivé nejdůležitější ukázky zdrojového kódu aplikace, které byly ovšem ještě zjednodušeny pro co nejnázornější ukázku řešení bez ohledu na celkový význam v programu. Celý kód je k dispozici v příloze.

Vytvoření a editace fontu

Tato ukázka znázorňuje získání dostupných fontů a určuje velikosti jejich písma:

```
for (int i = 0; i < System.Drawing.FontFamily.Families.Length;i++){
comboBox2.Items.Add(System.Drawing.FontFamily.Families[i].Name);}
for (int i = 4; i <= 12; i++){
comboBox1.Items.Add(i.ToString()}
```

Zde se určují horní a dolní meze jednotlivých znaků vybraného fontu:

```
for (i = 33; i <= 127; i++){
c = (char)i;
c2 = Convert.ToString(c);
g.Clear((Color.Empty));
g.DrawString(c2, new Font(comboBox2.Text, Convert.ToInt32(comboBox1.Text)),
new SolidBrush(Color.Black), 0, 0);
for (y = 0;y < OffsUp; y++){
for (x = 0; x < 100; x++){
Pixel = myBitmap.GetPixel(x, y);
if (Pixel.A == 255){
if (y<OffsUp) OffsUp = y;
break;}}}
for (y = 99; y >=OffsDwn; y--){
for (x = 0; x < 100; x++){
Pixel = myBitmap.GetPixel(x,y);
if (Pixel.A == 255){
if (y>OffsDwn) OffsDwn = y;
break;}}}
```

V tomto úseku kódu probíhá nastavení mezery mezi znaky pomocí písmen B a C :

```
g.DrawString("BC", new Font(comboBox2.Text, Convert.ToInt32(comboBox1.Text)), new
SolidBrush(Color.Black), 0, 0);
for (x = 0; x < 100; x++){  //zjisteni zacatku pismene B
for (y = 0; y < 100; y++) if ((Pixel2=(int)myBitmap.GetPixel(x, y).A ) == 255) break;
if (Pixel2 == 255) break;}
for (; x < 100; x++){  //zjisteni konec pismene B
for (y = 0; y < 100; y++) if ((Pixel2 = (int)myBitmap.GetPixel(x, y).A) == 255) break;
if (Pixel2 == 0) break;}
OffsMezera = x;
```

```
for (; x < 100; x++){  //zjisteni mezery, zacatek druheho pismene
for (y = 0; y < 100; y++)
if ((Pixel2 = (int)myBitmap.GetPixel(x, y).A) == 255) break;
if (Pixel2 == 255) break;}
OffsMezera = x - OffsMezera;
```

Úsek kódu pro zjištění levých a pravých mezí znaků fontu:

```
for (x = 0; x < 100; x++){  //zjisteni zacatku pismene, (ofset z leva)
for (y = 0; y < 100; y++) if ((Pixel2 = (int)myBitmap.GetPixel(x, y).A) == 255) break;
if (Pixel2 == 255) break;}
OffsLeft = x;
for (x = 99; x > 0; x--){  //zjisteni konce pismene, (ofset z prava)
for (y = 0; y < 100; y++) if ((Pixel2 = (int)myBitmap.GetPixel(x, y).A) == 255) break;
if (Pixel2 == 255) break;}
OffsRight = x;
```

Ukázka Editace fontu a jeho uložení do binárního souboru:

```
for (x = OffsLeft; x <= OffsRight; x++){
CountByte++;
ROT = 128;
for (y = OffsUp; y <= OffsDwn; y++){
if (ROT == 0){
CountByte++;
ROT = 128;}
if (myBitmap.GetPixel(x, y).A == 255){
Data[CountByte] = Convert.ToByte((int)Data[CountByte] | (int)ROT);}
ROT = Convert.ToByte((int)ROT >> 1);}}
CountByte++;
proud.Write(Data, 0, CountByte+1);
```

Vytvoření a editace textu z fontu

Část kódu pro získání napsaného textu:

```
CharString = richTextBox2.Text.ToCharArray();
SizeChar= CharString.Length;
for (i=0;i<SizeChar;i++){
if (CharString[i] == ' ') AdressChar=1;
else{znak = CharString[i];
if ((int)znak >= 32 && (int)znak <= 127){
Cbyte = (int)znak;
AdressChar = ((int)(Header[(Cbyte - 33) * 2 + 4])) << 8;
AdressChar += (int)Header[(Cbyte - 33) * 2 + 5];}
else {MessageBox.Show("Znaky textu musi byt v rozsahu ASCII od 32 do 127.");
```

```
return;}}
Cbyte=Data[AdressChar];
```

Ukázka editace vytvořeného textu a jeho uložení do binárního souboru:

```
while((Cbyte=(int)Data[++AdressChar])!=170){
DTImage[j++] = Convert.ToByte(Cbyte);}
CharString = null;
SizeDTImage = j;
proud.Write(DTImage, 0, SizeDTImage);
proud.Close();
```

Editace již vytvořeného obrázku:

Úsek kódu pro načtení vytvořeného obrázku:

```
OpenFileDialog OpenIMG = new OpenFileDialog();
LoadIMG = (Bitmap)Bitmap.FromFile(OpenIMG.FileName);
pictureBox1.Image = Bitmap.FromFile(OpenIMG.FileName);
```

Ukázka zdrojového kódu pro editaci načteného obrázku a jeho uložení do souboru:

```
for (x = 0; x < LoadIMG.Width; x++){
Line = 128;
for (y = 0; y < 8; y++){
if (LoadIMG.GetPixel(x, y).R == 0) DTImage[x * 2] |=
Convert.ToByte(Line);
Line=Line>>1;}
Line = 128;
for (y = 8; y < 16; y++){
if (LoadIMG.GetPixel(x, y).R == 0) DTImage[x * 2 + 1] |= Convert.ToByte(Line);
Line = Line >> 1;}}
SizeDTImage = LoadIMG.Width*2;
proud.Write(DTImage, 0, SizeDTImage);
proud.Close();
```

Animace editovaných dat

Část kódu pro animování editovaných fontů, textů nebo obrázků:

```
bool[] MaticeImage = new bool[32 * 16];
Graphics GraphicsBN = Graphics.FromImage(BitmapNahled);
GraphicsBN.Clear(Color.Empty);
for (x = 0; x < 32; x++){
for (y = 0; y < 16; y++){
if (MaticeImage[x * 16 + y]) GraphicsBN.FillRectangle(new
SolidBrush(Color.Red), x * 16 + 1, y * 16 + 1, 13, 13);
else GraphicsBN.FillRectangle(new SolidBrush(Color.Black), x * 16 + 1, y * 16 + 1, 13, 13);}}
```

```
pictureBox2.Image = BitmapNahled;}
```

Spuštění animace a zobrazení nápovědy

Úsek kódu pro spouštění animace:

```
if (!TimerON){
TimerON = true;
timer1.Start();}
else{
timer1.Stop();
TimerON = false;
```

Část programu pro zobrazení okna s nápovědou:

```
Form Help = new Help();
Help.Show();
```

Odesílání editovaných dat:

Ukázka kódu pro nalezení všech dostupných sériových portů:

```
allPorts = SerialPort.GetPortNames();
BoxPort.Items.Add(port);
```

Část kódu pro výběr portu a souboru:

```
OpenFileDialog DialogOpen = new OpenFileDialog();
proud.Read(Data, 0, LengthFile);
string NameFile = DialogOpen.FileName;
FileStream proud;
proud = new FileStream(@NameFile, FileMode.Open, FileAccess.Read);
LengthFile=(int)(proud.Length);
proud.Read(Data, 0, LengthFile);
proud.Close();
```

Úsek zdrojového kódu pro odeslání dat z binárního souboru:

```
serialPort1.PortName = BoxPort.Text;
serialPort1.Open();
byte[] B = new byte[2];
B[0]=Convert.ToByte((LengthFile - 2) >> 8);
B[1] = Convert.ToByte((LengthFile - 2) - (B[0] * 256));
byte[] StartUDR = new byte[1];
StartUDR[0] = 23;
serialPort1.Write(StartUDR, 0, 1);
serialPort1.Write(B, 0, 2);
serialPort1.Write(Data, 1, 1);
serialPort1.Write(Data, 2, LengthFile - 2); serialPort1.Close();
```

12.2 Program pro mikroprocesor

Program pro mikroprocesor AVR ATmega16 byl vytvořen v prostředí AVR Studio v programovacím jazyce C. Programovací jazyk C dosáhl velké obliby zejména proto, že má nejenom vlastnosti, které očekáváme od vyšších programovacích jazyků, ale i vlastnosti, které očekáváme spíše u Asembleru. Z vyšších programovacích jazyků má jazyk C nejblíže k hardwaru. Proto se i u velkých počítačů používá při vytváření operačních systémů. Používání vyšších programovacích jazyků respektují dokonce i tvůrci procesorů, když navrhují jejich jádro optimalizované pro práci s kompilátory nějakého vyššího jazyka. Také mikroprocesory ATMEL AVR jsou navrhovány tak, aby vyhovovaly zejména široce používanému programovacímu jazyku C.

Tento software je životně důležitý pro celou funkci řídicí aplikace, neboť bez něj by mikroprocesor nevykazoval žádnou aktivitu. V programu je obsažen zdrojový kód, který v podstatě říká procesoru jak má používat své vybavení pro řízení celé aplikace.

[19]

12.2.1 Popis funkce softwaru pro MCU

Obr. 30 znázorňuje strukturu a funkci celého programu. Základ tvoří jednotlivé funkce, které realizují dílčí kroky v programu.

Program tedy po zapnutí celého zařízení a úvodní inicializaci zahájí čtení dat, která jsou uložena v paměti EEPROM. Samotné čtení je realizováno touto funkcí :

- void READ_EEPROM_TO_BUFFER(void);

Čtení dat z EEPROM se na začátku provádí proto, aby mohlo zařízení hned vykreslovat, pokud jsou již data do EEPROM zapsána. V opačném případě se nevykreslí nic a uživatel musí do EEPROM data zapsat.

Po přečtení dat z EEPROM systém čeká na impuls, který je směrodatný pro jeho další činnost. Může to být signál buďto od mikrospínače pro přenos a zápis dat do EEPROM (INT0) nebo od Hallova senzoru (ICP) pro zobrazování.

Jestliže příjde na vstup MCU signál z mikrospínače, zahájí se tím režim přijímání dat pomocí rozhraní USART a žádné další činnosti kromě této nejsou povoleny. Při vyvolání přerušení od USART se tedy data ukládají do pomocné proměnné, která je pak po ukončení přenosu dat použita pro zápis do EEPROM.

Z PC byly pomocí Editačního uživatelského programu zároveň kromě zdrojových dat potřebných k vykreslení obrazce také odeslány rozlišovací informace, které by ovšem jinak mohly narušit celistvost vykreslovaného obrazce. Tyto rozlišovací informace určují, zda- li se jedná o obrázek nebo o text, jestli zahrnují dva řádky nebo jeden (8 LED nebo 16) a zda bude vykreslování probíhat se statickým nebo dynamickým obrazem. Proto se v obsluze přerušení pro USART nejprve určí velikost přenášeného souboru, a protože jsou to vždycky první dva bajty jsou následně při čtení dat z EEPROM odděleny od dat vykreslovacích.

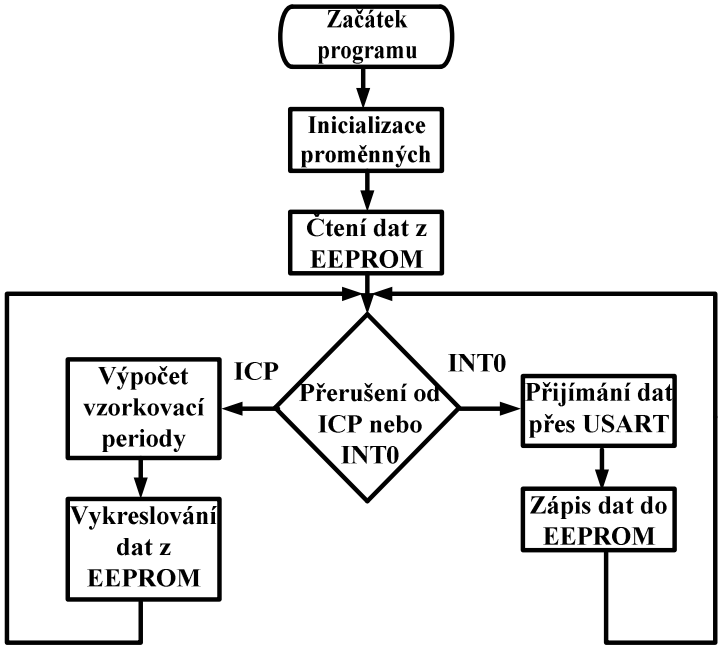
Po ukončení přenosu dat se tyto data zapíší do paměti EEPROM pomocí funkce:

- void WRITE_BUFFER_TO_EEPROM(void);

Po zápisu jsou data znovu přečtena a mikroprocesor čeká na další vstupní impuls opět buď od mikrospínače nebo od Hallova senzoru.

Pokud tedy dostane mikroprocesor na svůj vstup impuls od Hallova snímače (což znamená chod motoru), započne se výpočet časové konstanty t_s. Tento výpočet tkví ve využití 16- ti bitového časovače mikroprocesoru. Při vyvolání přerušení od Hallovy sondy se hodnota, která je uložena v registru tohoto časovače v hlavní funkci main() přepočítá a následně díky funkci _delay jsou zobrazovací data posílána na příslušné výstupy mikroprocesoru v čase, čímž se vykreslí jeden vzorek obrazce. Časový impuls uložený v registru čítače lze také ovlivnit nastavením děličky kmitočtu. Tím se dosáhne jiné délky jednoho impulsu který čítač načte. Časová konstanta je také ovlivněna otáčkami motoru.

Po ztrátě signálu z Hallova snímače otáček (motor se netočí) čeká systém na další impuls.



Obr. 30 Vývojový diagram softwaru pro mikroprocesor

12.2.2 Princip výpočtu časové konstanty pro vykreslení sloupce s LED

Jak již bylo uvedeno, základem pro výpočet časové konstanty je 16- ti bitový čítač/ časovač, do jehož registru se ukládají impulsy o délce dané nastavením děličky kmitočtu rezonátoru mikroprocesoru. Princip je tedy takový, že na základě vyvolaného přerušení od Hallova snímače otáček se počet impulsů uložených v registru časovače vynásobí s odvozenou dobou trvání jednoho impulsu a to celé se následně podělí počtem vykreslovaných sloupců podle rozlišení (64). V Tab. 6 jsou uvedeny příklady hodnot otáček a vypočtené časové konstanty mikroprocesorem. Z tabulky je zřejmé, že jsou výsledky výpočtu časové konstanty

mikroprocesorem téměř shodné s výsledky vypočtených teoreticky v kapitole 9 v Tab. 5. Mikroprocesor nevyhodnotí desetiny impulsu, pouze celé impulsy.

Zde je uveden příklad výpočtu časové konstanty mikroprocesorem s vloženým zdrojovým kódem:

$$n = 2000 ot./\min. \Rightarrow t_n = 30 ms$$

$$f_r = 8 MHz$$

$$f_d = \frac{f_r}{1024} = \frac{8 \cdot 10^6}{1024} = 7812{,}5 Hz$$

$$t_i = \frac{1}{f_d} = \frac{1}{7812{,}5} = \underline{128 \mu s}$$

$$n_i = \frac{t_n}{t_i} = \frac{30 \cdot 10^{-3}}{128 \cdot 10^{-6}} = \underline{234}$$

$$t_s = n_i \cdot \frac{t_i}{64} = 234 \cdot \frac{128 \cdot 10^{-6}}{64} = \underline{\underline{468 \mu s}}$$

$$\frac{t_i}{64} = \frac{128 \cdot 10^{-6}}{64} = 0{,}002 ms$$

TML=(double)(TCNT1*0.002);

n	t_n	t_s
[min.]	[ms]	[μs]
500	120	1874
800	75	1170
1000	60	936
1500	40	624
2000	30	468
2300	26	406
2500	24	374
3000	20	312
3500	17	264

Tab. 6 Příklady mikroprocesorem vypočtených hodnot t_s pro různé hodnoty otáček

12.2.3 Ukázky nejdůležitějšího zdrojového kódu pro software mikroprocesoru

Tato část podkapitoly ukazuje nejdůležitější a zkrácené úseky zdrojového kódu programu.

Příjem dat pomocí USART:

Část kódu k vykonání obslužné rutiny USART-příjem dat:

```
TUDR=(unsigned char)UDR;
if (StartUDR==23){
if (buffCt==0){
length_buffer = (unsigned int)(TUDR << 8) ;
buffer[ buffCt ] = TUDR;
buffCt++;}
else if (buffCt==1){
length_buffer += (unsigned int)(TUDR) + 2;
buffer[ buffCt ] = TUDR;
buffCt++;}
else {
buffer[ buffCt ] = TUDR;
buffCt++;
if (buffCt>length_buffer) ENDRU=0;}
else   StartUDR=(unsigned int)TUDR;
```

Zápis do EEPROM:

```
unsigned char   temp;
for (i=0;i<buffCt;i++){
while(EECR & (1<<EEWE))
EEAR = i;
EEDR = buffer[i];
```

Čtení dat z EEPROM:

```
while(EECR & (1<<EEWE))
EEAR = 0;
buffer[0] = EEDR;
EEAR = 1;
buffer[1] = EEDR;
length_buffer = (unsigned int)(buffer[0]  << 8) ;
length_buffer += (unsigned int)(buffer[1]) + 2;
for (i=2;i<=length_buffer;i++){
EEAR = i; buffer[i] = EEDR;}}
```

Přepočet časové konstanty pro přenější provredení funkce delay:

```
while(1){
if (START==1 && SFLAG==1){
for (i=3;i<130;i=i+2){
temp=TML; // udaj z registru casovace
TMP1=buffer[i];
TMP2=buffer[i+1];
PORTA=TMP1;
PORTC=TMP2;
tst=((int)temp*1000);
tst=((int)tst%100);
temp=temp-(double)tst/1000;
_delay_ms(temp);
_delay_us(tst);}¨
PORTC= 0;PORTA= 0;}
```

13 Závěr

Cílem této bakalářské práce bylo vytvořit vestavěný řídicí systém pro aplikaci Dynamického zobrazovače. Po seznámení s danou problematikou byly určeny všechny požadavky na celý systém a následně byly navrženy způsoby řešení jednotlivých konstrukčních i softwarových požadavků, které se pak postupně realizovaly.

Jedním ze základních problémů při návrhu byla tvorba konstrukce, která je jedním z nejdůležitějších článků celého návrhu, neboť teprve po návrhu konstrukce bylo možné navrhnout další části systému, které jsou právě konstrukcí velmi omezeny.

V současné době je tedy realizována aplikace Dynamického zobrazovače, která kromě samotného zařízení ještě obsahuje uživatelský program v PC pro vytváření libovolných textů a obrazců (obrázky musí být černobílé a v rozlišení 16 x 64 pixelů), které se převádějí do digitální formy dat, vhodné pro zobrazování. Tyto data se pak pomocí tohoto uživatelského rozhraní odesílají z PC do Dynamického zobrazovače. Momentálně je tedy aplikace schopna přijímat data z PC a na základě těchto dat vykreslovat příslušný obrazec či text, takže se dá říci., že je celý systém funkční a do určité míry splňuje všechny požadavky, které jsou na něj kladeny.

Chod aplikace je znázorněn v jednotlivých přílohách, ze kterých jsou patrné i některé nedostatky, např. rozmístění zobrazovacích LED, které jsou ještě stále celkem daleko od sebe a výsledný obraz není tolik kompaktní. Také vlivem jejich zakřivení je obraz v horní části mírně zdeformovaný, což dělá problém zejména při vykreslování textu, ale na druhou stranu při vykreslování obrazců to působí efektnějším dojmem. Jedním z dalších nedostatků je občasný nárůst nebo úbytek nastavených otáček pomocí PWM. Tento problém pravděpodobně způsobuje nestejnoměrné nabíjení a vybíjení kondenzátoru, který určuje střídu PWM. Nestejnoměrné nabíjení tohoto kondenzátoru by mohlo být způsobeno rozdílem v dynamických odporech použitých diod pro stejné nabíjení i vybíjení kondenzátoru. V uživatelském programu není vhodné vybírat nejmenší velikosti písma pro tvorbu fontu, protože data jsou v tomto

případě převedena chybně. Problémem je pravděpodobně velmi malá velikost samotného písma, neboť je font editován jako bitmapa o určitých rozměrech a program již nedokáže takto malé znaky v poměrně velkém prostoru úspěšně převádět.

14 Použitá literatura

[1] *Lidské oko* [online]. PanWiki, [cit. 2010 08 14].URL: http://panwiki.panska.cz/index.php/Lidsk%C3%A9_oko.

[2] MACHÁČEK, Z. *Základy teorie signálů a soustav.* Studijní texty. VŠB- TUO. URL: <http://www.fei1.vsb.cz/stud_mat/K450/>.

[3] MACHÁČEK, Z., NEVŘIVA, P. *Číslicové signály a soustavy.* Studijní texty. VŠB-TUO. URL: <http://www.fei1.vsb.cz/stud_mat/K450/>.

[4] *Shannonův teorém* [online]. Wikipedie Otevřená Encyklopedie, [cit. 2010 08 14]. URL http://cs.wikipedia.org/wiki/Shannon%C5%AFv_teor%C3%A9m.

[5] KOTZIAN, J. *Senzory a měření.* Studijní texty. VŠB- TUO. URL: <http://www.fei1.vsb.cz/stud_mat/K450/>.

[6] Katalogové listy. *TLE4905L.*. [online], GES Electronics, spol. [cit. 2010-08-14]. URL: <http://www.ges.cz>.

[7] MATOUŠEK, D. *USB prakticky s obvody FTDI.* Praha: BEN, 2003. ISBN 80-7300-103-9.

[8] *O firmě FTDI* [online]. Asix server, [cit. 2010 08 14]. URL: http://www.asix.cz/ftabout.htm.

[9] Katalogové listy. *FT232RL.* [online], GM Electronic, spol. [cit. 2010-08-14]. URL: <http://www.gme.cz>.

[10] *Sériová linka RS232*[online]. HW server, [cit. 2010 08 14]. URL: http://hw.cz/rs-232.

[11] *Atmel* [online]. Wikipedie Otevřená Encyklopedie, [cit. 2010 08 14]. URL: http://cs.wikipedia.org/wiki/Atmel

[12] Katalogové listy. *AVR ATmega16.* [online], GM Electronic, spol. [cit. 2010-08-14]. URL: <http://www.gme.cz>.

[13] BEZDĚK, M. *Elektronika I.* České Budějovice: Koop, 2003. ISBN 80-7232-171-4.

[14] VOŽENÍLEK, L., LSTIBUREK, F. *Základy elektrotechniky II*. Praha: SNTL1985[15]

[15] *NE555* [online]. Wikipedie Otevřená Encyklopedie, [cit. 2010 08 14]. URL: http://cs.wikipedia.org/wiki/NE555.

[16] Katalogové listy. *ULN2803A.* [online], GM Electronic, spol. [cit. 2010-08-14]. URL: <http://www.gme.cz>.

[17] Katalogové listy. *7805 DPAK SMD.*. [online], GM Electronic, spol. [cit. 2010-08-14]. URL: <http://www.gme.cz>.

[18] *PWM regulátor* [online]. [cit. 2010 08 14]. URL: http://www.pokusy.chytrak.cz/schemata/wmp.htm

[19] LÁNÍČEK, R. *Elektronika- obvody, součástky, děje*. Praha: BEN, 1998. ISBN 80-86056-25-2.

[20] VÁŇA, V. *Mikrokontroléry Atmel AVR- Programování v jazyce C* . Praha: BEN, 2003. ISBN 80-7300-102-0.

15 Seznam příloh

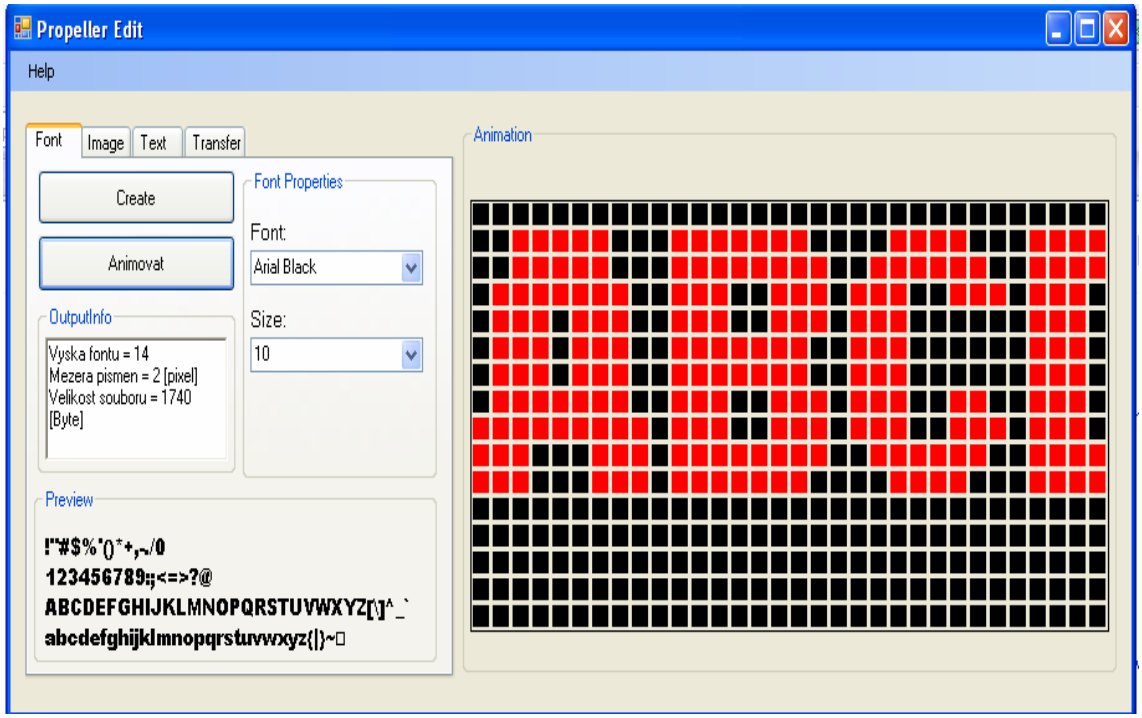
Příloha 1 Screenshoty Editačního uživatelského programu

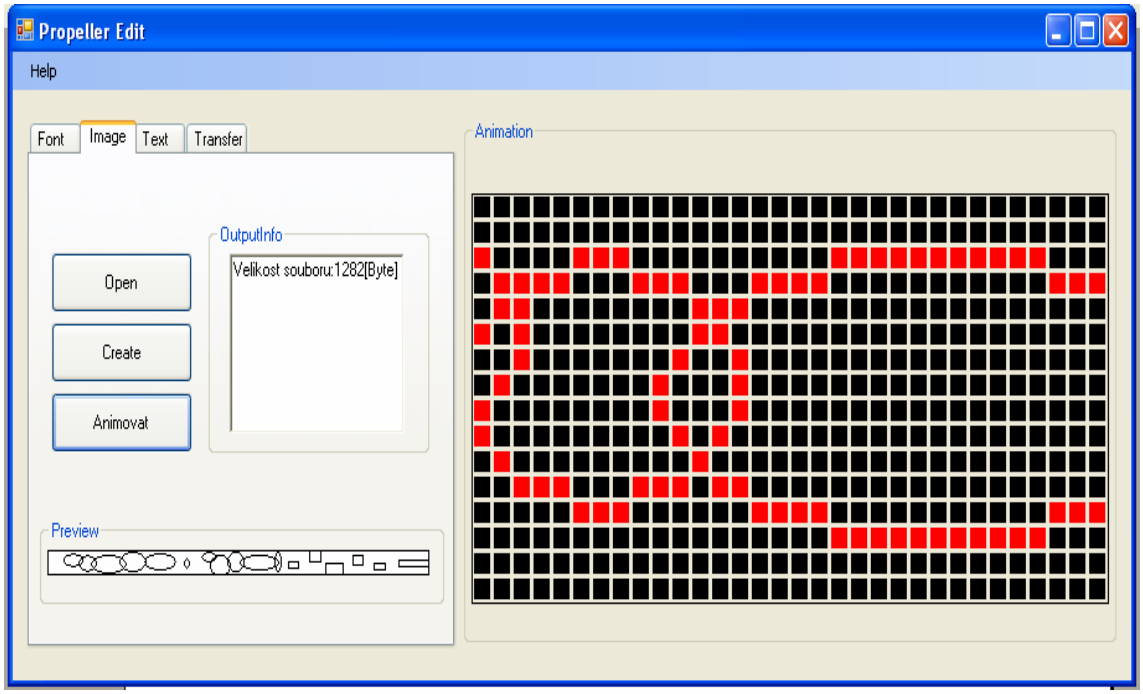
Příloha 2 Fotografie výrobku

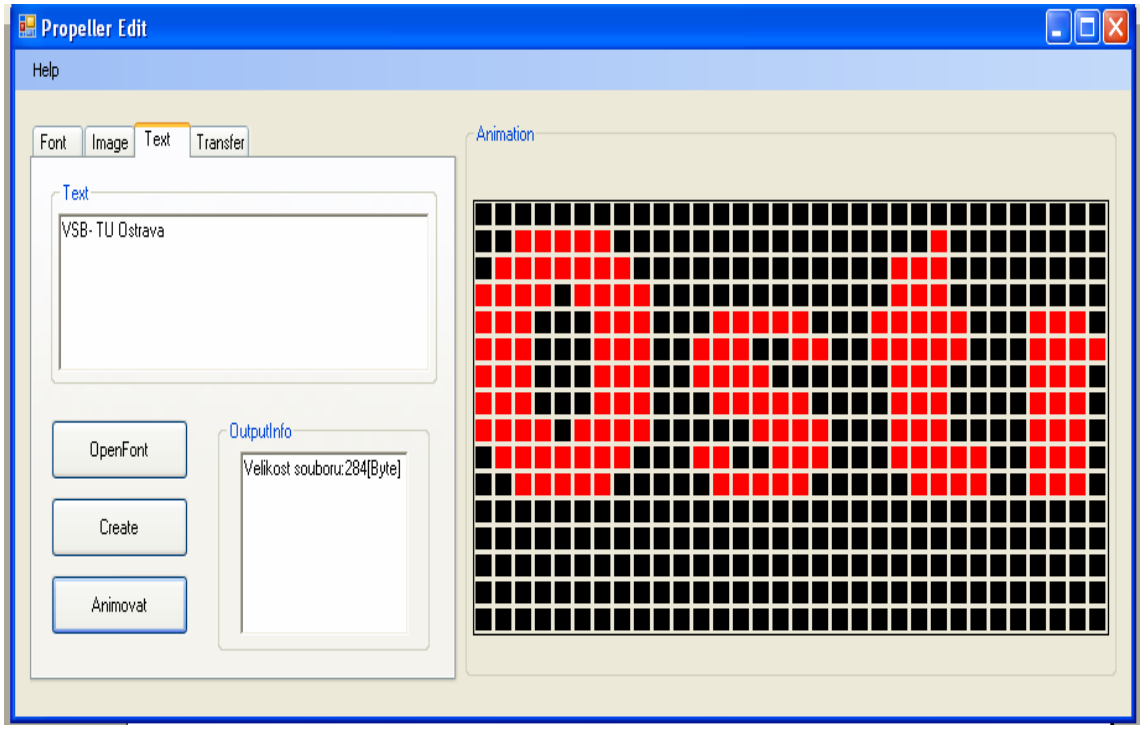
Příloha 3 CD s kopií této bakalářské práce

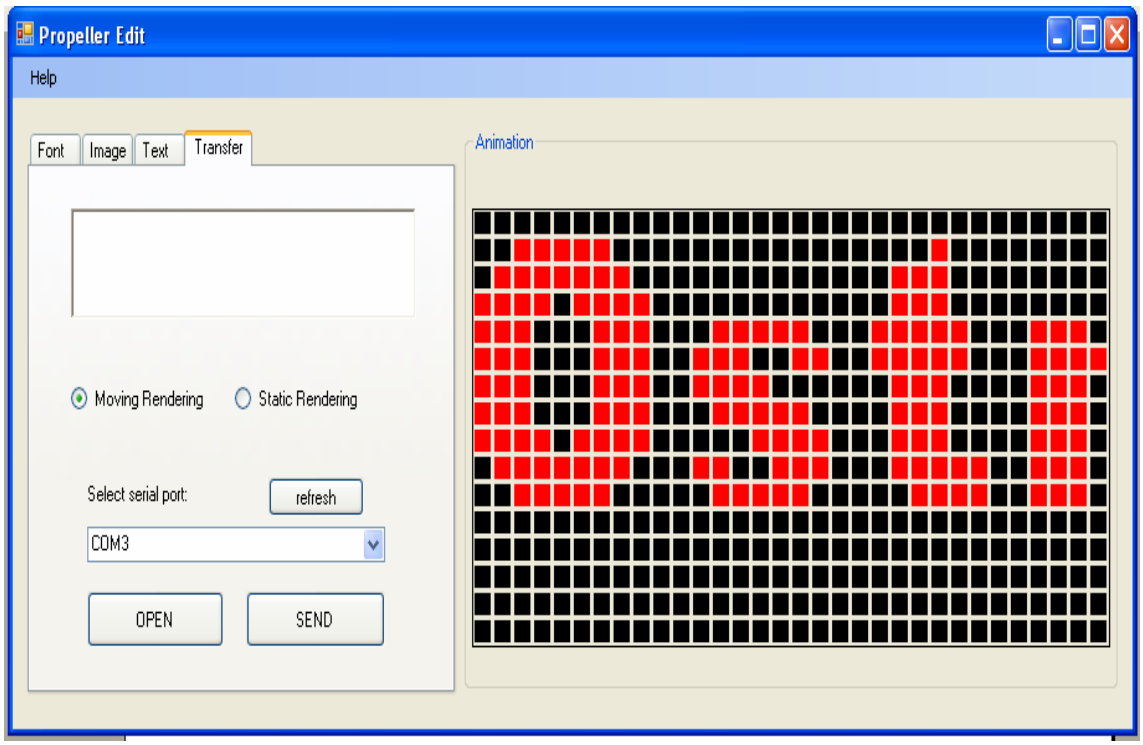
Příloha 1

Screenshoty Editačního uživatelského programu



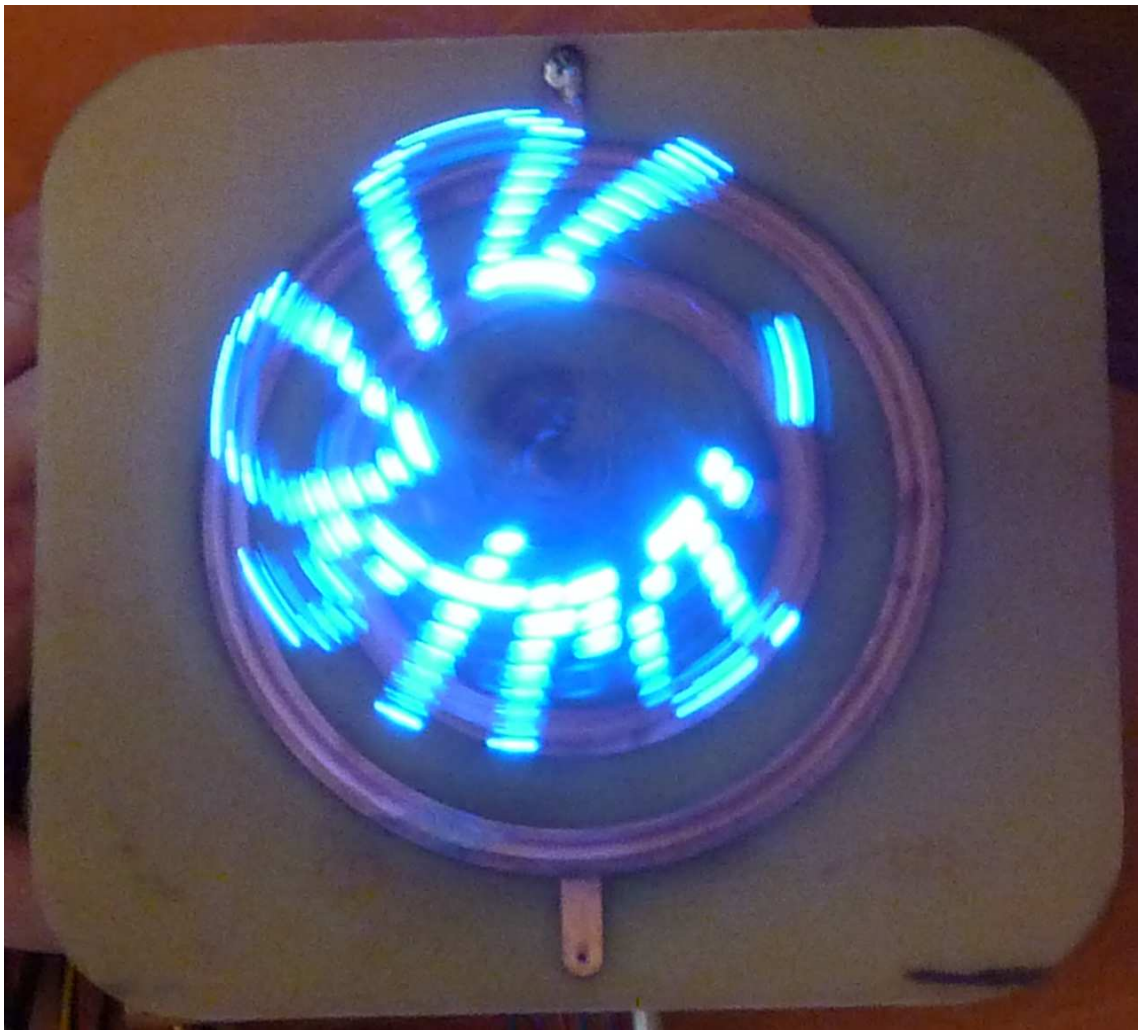


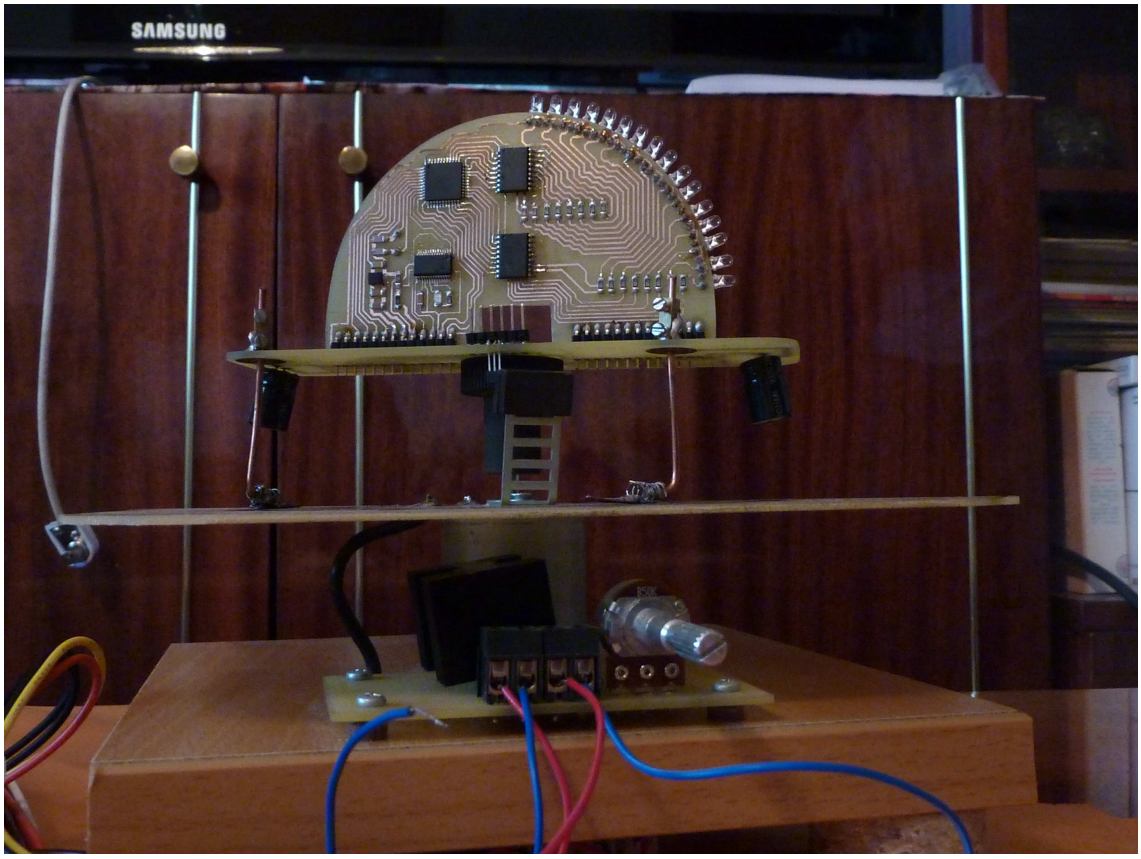

Propeller Edit
Help
Font
Image
Text
Transfer
Text
VSB- TU Ostrava
OpenFont
Create
Animovat
OutputInfo
Velikost souboru:284[Byte]
Animation


Propeller Edit
Help
Font
Image
Text
Transfer
Moving Rendering
Static Rendering
Select serial port:
refresh
COM3
OPEN
SEND
Animation

Příloha 2

Fotografie výrobku




SAMSUNG